ARTICLE IN PRESS

ELSEVIER

Decision Support Systems 1035 (2002) xxx–xxx

Decision Support Systems

www.elsevier.com/locate/dsw

A function-decomposition method for development of hierarchical multi-attribute decision models

Marko Bohanec[a,b,*], Blaž Zupan[c,d]

[a] *Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia*
[b] *University of Ljubljana, School of Public Administration, Ljubljana, Slovenia*
[c] *University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia*
[d] *Baylor College of Medicine, Department of Molecular and Human Genetics, Houston, TX, USA*

Received 2 August 2001; received in revised form 1 June 2002; accepted 5 August 2002

Abstract

Function decomposition is a recent machine learning method that develops a hierarchical structure from class-labeled data by discovering new aggregate attributes and their descriptions. Each new aggregate attribute is described by an example set whose complexity is lower than the complexity of the initial set. We show that function decomposition can be used to develop a hierarchical multi-attribute decision model from a given unstructured set of decision examples. The method implemented in a system called HINT is experimentally evaluated on a real-world housing loans allocation problem and on the rediscovery of three hierarchical decision models. The experimentation demonstrates that the decomposition can discover meaningful and transparent decision models of high classification accuracy. We specifically study the effects of human interaction through either assistance or provision of background knowledge for function decomposition, and show that this has a positive effect on both the comprehensibility and classification accuracy.

Keywords: Multi-attribute decision making; Hierarchical models; Function decomposition; Data-driven modeling; Data mining

1. Introduction

In decision support systems, we use models to predict the outcome of decision choices we might make [13]. For multi-attribute decision problems, that is, decision-making situations in which the alternatives are described by several conflicting attributes, we develop multi-attribute decision models. Most commonly, such models are developed in a hierarchical fashion, starting from some general but imprecise goal statement, which is gradually refined into more precise sub and sub-sub goals [28]. A typical example of this approach is Saaty's [22] analytic hierarchy process, but there are also many others [5,8,30].

The development of hierarchical multi-attribute decision models is difficult, especially when the decision problem itself is complex and involves several tens of attributes. In most cases, the models are developed manually in a tiresome and lengthy process in which the designers (decision analysts, decision makers, experts, knowledge engineers) use their knowledge about the problem and employ their skill and experience. Computers provide relatively inexpensive and available means to collect data, and there is a growing volume of data about

* Corresponding author. Jožef Stefan Institute, Jamova 39, SI-1000 Ljubljana, Slovenia. Tel.: +386-1-477-3309; fax: +386-1-425-1038.

E-mail address: marko.bohanec@ijs.si (M. Bohanec).

0167-9236/02/$ - see front matter © 2002 Elsevier Science B.V. All rights reserved.
PII: S0167-9236(02)00148-3

decisions already made. This data may contain useful information for decision support and discovery of underlying decision models through data mining [10,27].

In this paper, we show that function decomposition, an approach originally developed to assist in the design of digital circuits [1,7], and recently extended for its use in machine learning and data mining [32], can be used to develop hierarchical multi-attribute decision models from the set of labeled (evaluated) *examples*. Decision examples may be taken either from an existing database of past decisions, or may be provided explicitly by the decision maker. Each example is described by a set of attributes and its utility. The method is restricted to decision problems with nominal attributes and utility. Given an initial set of examples, the method develops a corresponding model in terms of a hierarchy of attributes and their definitions. The development proceeds either with or without human interaction.

The method we use, *function decomposition*, is based on recursive decomposition of initial set of decision examples, thus deriving smaller sets of examples with less attributes, and placing them into a decision model hierarchy. Let a set of decision examples E_F with *attributes* $X=\langle x_1,\ldots,x_n\rangle$ and *utility* variable y partially represent a utility function $y=F(X)$. The goal is to decompose this function into $y=G(A,H(B))$, where A and B are subsets of attributes such that $A\cup B=X$, and functions G and H are partially represented by sets of examples E_G and E_H, respectively. The task of decomposition is to determine E_G and E_H so that their complexity (determined by some complexity measure) is, if possible, lower than that of E_F, and so that E_G and E_H are consistent with E_F. Such a decomposition also discovers a new *aggregate attribute* (hereafter referred to as *concept*) $c=H(B)$. Since the decomposition can be applied recursively on E_G and E_H, the result in general is a *hierarchy* consisting of attributes (terminal nodes) and concepts (internal nodes). For each concept in the hierarchy, there is a corresponding set of examples (such as E_H) that describes the dependency of that concept on its immediate descendants in the hierarchy. Hence, the concept can be interpreted as a sub-goal within the goal hierarchy.

Central to each decomposition step is the selection of a partition of attributes X to sets A and B. This is guided by a *partition selection measure* that assesses the joint complexity of the resulting E_G and E_H. The decomposition selects the partition that minimizes this complexity measure. Although such decomposition can be completely autonomous, the comprehensibility of the discovered structure may be improved if the user is involved in the partition selection process: for instance, a few of the better, or less complex, partitions may be presented to the user, who selects the best candidate and assigns a label to the new concept.

Function decomposition was first introduced as a method for the development of digital circuits [1,7]. There, the decision examples were represented as a truth table with binary input variables for attributes and binary outcome variables for utility. Using several additional approaches, the method was recently extended to handle multi-valued attributes [32] and work over a set of decision examples that does not necessary cover the whole attribute space [29]. It was shown [32] that as such, function decomposition can be used as machine learning method that can discover meaningful concepts that generalize well.

In this paper, we position function decomposition within decision support framework, showing that the discovered models are essentially a hierarchical multi-attribute decision support models. These are, furthermore, in terms of their formal representation identical to those used by a decision support system called DEX [2]. We particularly study the effects of human assistance, and show that various types of human intervention in the process of function decomposition may have positive effects both on the quality of resulting hierarchy and accuracy of the model. Overall, the experimental assessment in the paper shows that function decomposition (1) may discover meaningful concepts that are described through (2) small and manageable sets of examples and that are placed in a (3) meaningful hierarchy that (4) generalizes well to the alternatives not considered in the original set of decision examples.

The paper is organized as follows. Section 2 introduces basic concepts of the decomposition method. In Section 3, the method is experimentally evaluated on a real-world problem of housing loans allocation. The issues specifically addressed are: comprehensibility, the benefit of user's interaction in the decomposition process, classification accuracy of the developed model, and its relation to common

multi-criteria models. These issues are further studied in Section 4, where function decomposition is tested on the rediscovery of three hierarchical decision models of moderate complexity. Section 5 overviews the related work. The paper concludes with a discussion and summary.

2. Decomposition method

This section briefly introduces the decomposition method. For a more elaborate and formal description of the method, the reader is referred to Ref. [32]. First, we introduce a single-step decomposition, which, given a set of decision examples E_F, decomposes it to consistent sets E_G and E_H. This is followed by the description of overall decomposition algorithm that, given an initial set of examples, iteratively applies the single-step decomposition to derive a hierarchy of concepts. Each concept in the hierarchy is described by its own set of examples. In each iteration, the decomposition algorithm also deals with the problem of attribute selection.

2.1. Single-step decomposition

The core of the decomposition algorithm is a *single-step decomposition* which, given a set of examples E_F that partially specifies a function $y=F(X)$, and a partition of attributes X to sets A and B (denoted $A|B$), decomposes F into $y=G(A,c)$ and $c=H(B)$. This is done by constructing sets of examples E_G and E_H that partially specify G and H, respectively. X is a set of attributes $x_1,\ldots,x_m$, and c is a new concept. The partition $A|B$ is composed of a *free* set A and *bound* set B such that $A \bigcup B = X$ and $A \cap B = \varnothing$. E_G and E_H are developed in the decomposition process and are not predefined in any way.

Table 1
Set of examples E_F that partially describes the function $y=F(x_1,x_2,x_3)$

x_1	x_2	x_3	y
lo	lo	lo	lo
lo	lo	hi	lo
lo	med	lo	lo
lo	med	hi	med
lo	hi	lo	lo
lo	hi	hi	hi
med	med	lo	med
med	hi	lo	med
med	hi	hi	hi
hi	lo	lo	hi
hi	hi	lo	hi

Table 2
Partition matrix with column labels (c) for examples from Table 1 using the attribute partition $\langle x_1\rangle|\langle x_2,x_3\rangle$

	x_2	lo	lo	med	med	hi	hi
x_1	x_3	lo	hi	lo	hi	lo	hi
lo		lo	lo	lo	med	lo	hi
med		–	–	med	–	med	hi
hi		hi	–	–	–	hi	–
c		1	1	1	2	1	3

Let us describe the single-step decomposition through an example. Suppose there is a set E_F (Table 1) that partially describes a function $y=F(x_1,x_2,x_3)$, where x_1, x_2, and x_3 are attributes and y is the target concept. The variables y, x_1, and x_2 can take the values lo, med, hi, and x_3 can take the values lo, hi.

Suppose that we want to derive the example sets E_G and E_H for the attribute partition $A|B=\langle x_1\rangle|\langle x_2,x_3\rangle$. For this purpose, the initial set of examples is first represented by a *partition matrix*, which is a tabular representation of E_F with all combinations of values of attributes in A as row labels and of B as column labels. Each example from E_F has its corresponding entry in the matrix. Partition matrix entries with no corresponding example in E_F are denoted by '–' and treated as *don't-cares*: unobserved entries that can, in principle, be assigned any value during the decomposition process. For our example set (Table 1) and the above attribute partition, the partition matrix is given in Table 2.

Each column in the partition matrix denotes the behavior of F when the attributes in the bound set are constant. Columns that exhibit the same behavior, that is, have pairwise equal row entries, or at least one row entry is a don't-care, are called *compatible*. Compatible columns can be labeled with the same value of c, a concept which will have some value to the decision maker independent of the attributes in the free set (x_1 in our case). For instance, the first two columns in Table 2 are compatible: their entries in the

first row are equal, and at least one entry in the remaining rows is a don't-care.

Single-step decomposition aims at deriving the new concept variable c having the smallest set of possible values. For that purpose, we construct an *incompatibility graph* whose vertices correspond to partition matrix columns. Two vertices are connected if the corresponding columns are not compatible. To find the labels of c, the incompatibility graph is colored: the coloring will assign different labels to the vertices that represent mutually incompatible columns, while the compatible columns may share the same color.

An optimal coloring identifies the minimal number of groups of mutually compatible columns. This number is called *column multiplicity* and denoted by $v(A|B)$. Column multiplicity equals to the lowest number of possible values to be used for the new concept variable c. Since optimal graph coloring is an NP-hard problem, we use a heuristic method of polynomial complexity [16]. For our example partition matrix, the incompatibility graph is given in Fig. 1. Notice that three colors are required to color this graph.

After the coloring of the incompatibility graph, each column of the partition matrix is assigned a label, which corresponds to an abstract value of the new concept variable c. From such an annotated partition matrix, the new sets E_G and E_H can be derived. For E_H, the attribute set is B. Each column in partition matrix is an example in E_H. The label

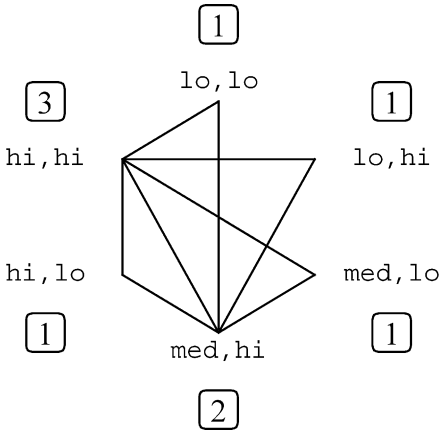

Fig. 1. Incompatibility graph with assigned colors (labels) for the partition matrix from Table 2.

(color) of the column becomes the class value of that example.

E_G is derived as follows. For each value of c and combination of values of attributes in A, $y=G(A,c)$ is determined by looking for an example $e_i \in E_F$ in the corresponding row and in any column labeled with the value of c. If such an example exists, it is included in E_G using the attributes $A \cup \{c\}$ and class $y=F(e_i)$.

Fig. 2a shows E_G and E_H for the decomposition of Table 2. Notice that the new sets are less complex than the initial set E_F, and are thus much easier to interpret: it is easy to see that c corresponds to $\min(x_2,x_3)$ and y to $\max(x_1,c)$. The corresponding interpretation of the three possible values of c is: `1`=lo, `2`=med, `3`=hi.

There exist two other non-trivial partitions of the same attribute set: $\langle x_2\rangle|\langle x_1,x_3\rangle$ and $\langle x_3\rangle|\langle x_1,x_2\rangle$. The corresponding decompositions are shown in Fig. 2b and c, respectively. Compared to the decomposition using the partition $\langle x_1\rangle|\langle x_2,x_3\rangle$, they result in overall larger sets E_G and E_H and introduce intermediate concepts with more values (4 and 5, respectively, instead of 3). Moreover, the resulting sets are harder to interpret. Among the three attribute partitions, it is therefore preferable to choose the first one.

The decomposition algorithm can generalize. This means that it can define an unobserved entry ('–') in row a and column b if there exists a corresponding example e_i that belongs to row a and shares the same column label as b. For example, the entry in row `hi` and column ⟨lo,hi⟩ in Table 2 can be represented as `hi` because the column ⟨lo,hi⟩ has the same column label as columns ⟨lo,lo⟩ and ⟨hi,lo⟩. Note that the same logic implies that the '–' in column ⟨med,lo⟩ has a value of `hi` as well.

The single-step decomposition can also be used to detect redundant attributes. Let an initial set of attributes X be partitioned to $B=\langle x_j\rangle$ and $A=X\backslash\langle x_j\rangle$. If $v(A|B)=1$, then the corresponding function $c=H(x_j)$ is constant, and the attribute x_j can be removed from the set of examples.

2.2. Overall decomposition method

Given a set of examples E_F that partially defines a utility function $y=F(X)$, where $X=\langle x_1,\ldots,x_n\rangle$, it is particularly important to find an appropriate attribute partition $A|B$ of the set X for the single-step decomposition. As illustrated in the previous section, the

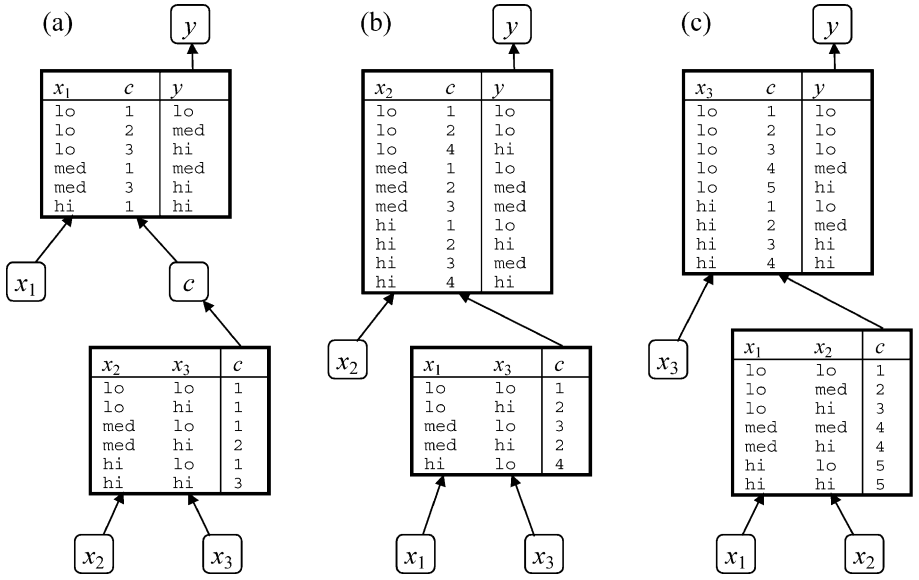

Fig. 2. Three different decompositions of data from Table 1.

partition selection can affect both the complexity and comprehensibility of the resulting example sets. Zupan [31] proposed several *partition selection measures* of which in this paper we mention and use only the simplest one: column multiplicity $v(A|B)$. In this case, the decomposition method favors the attribute partitions that yield the intermediate concepts with the smallest sets of possible values. For our example, this criterion prefers the decomposition from Fig. 2a, which indeed results in the simplest and most comprehensible description of the target concept.

To increase the tractability of the analysis, we propose to consider only partitions that feature candidate concepts with a limited number of attributes. In the experiments presented in Sections 3 and 4, partitions with only two or three attributes in the bound sets were investigated. Such a constraint results in intermediate concepts with only a few attributes, which may have a positive effect on comprehensibility and the size of generated sets of examples.

A single-step decomposition takes a set of examples and decomposes it to a system of two less complex but equivalent data sets. These two sets are then further investigated for decomposition. The overall decomposition is then a recursive process that results in a hierarchy of data sets.

The process of the decomposition may be moderated. The user may be involved in the selection of partitions and selection of the sets to decompose, or may impose some other constraints on the hierarchy to be developed. We refer to such a process to as *supervised decomposition*. Compared to *unsupervised* decomposition [31], where in the absence of user's involvement, the algorithm is responsible for choosing the best attribute partition, we expect the supervised decomposition to perform better in terms of comprehensibility, particularly in the cases where the initial examples sparsely cover the attribute space.

2.3. Implementation

The decomposition method was first implemented in the C language as a system called Hierarchy INduction Tool (HINT). The system runs on several UNIX platforms, including HP-UX, SGI Iris, and SunOS. Recently, HINT became a component of the data mining framework Orange (http://www.magix.fri.uni-lj.si/orange), which incorporates the scripting language Python to allow extensive experimentation and studies of different function decomposition scenarios. The definition of domain names and examples, and the guidance of the

decomposition as presented in this paper is therefore managed by Python scripts.

3. Case study: model discovery for housing loan allocation

The experimental evaluation of the method was carried out using a real-world database used in a management decision support system for allocating housing loans [3]. This system was developed for the Housing Fund of the Republic of Slovenia and has been used in 21 loan programs since 1991. The total loan amount allocated so far with the aid of this system corresponds to € 254 million, which is about two thirds of all housing loans approved in Slovenia in that period.

In each float, the principal problem is to allocate the available funds to applicants. Typically, there are several thousand applicants and the total requested amount exceeds the available financial resources. Therefore, the applicants must be ranked in a priority order for the distribution of resources in accordance with the criteria prescribed in the tender. Each applicant is ranked into one of five priority classes. The classes determine the amount of loan granted to the applicant: the higher the priority class, the higher the proportion of the request is granted to the applicant. Depending on the available funds, loans are not typically granted to applicants ranked in lower priority classes.

The Housing Found determines the loan priority by using a hierarchical multi-attribute decision model whose structure is presented in Fig. 3. It consists if three subtrees. The first subtree corresponds to applicant's present housing conditions (house), and includes: the stage of solving their housing problem (i.e., does an applicant already possess a housing property), the ownership and suitability of present housing, and the way of solving their housing problem (is the property declared a cultural or historical monument, do any other advantages apply to the application, and did the applicant obtain other financial sources from the government). The second subtree determines the applicant's status in terms of their earnings, employment and the number of children. The third subtree, soc_health, determines the priority according to the applicant's health status (e.g., higher priority for the disabled), family conditions (e.g., higher priority for families with children), and age (e.g., higher priority for young families).

For each internal concept in the structure, there is a decision rule defined for the aggregation of concepts. In the actual application, both the structure and the rules were developed manually by the experts using a multi-attribute decision making shell DEX [2].

For the evaluation of the decomposition method, we took applicants' data from one of the floats carried out in 1994. In addition to some general data, such as the name and address of the applicant, each data record contained 12 two- to five-valued attributes that were essential for the determination of loan priority. There were five two-valued, five three-valued and two five-valued attributes, so the size of the attribute space (i.e., the number of possible unique input vectors) was $2^5 3^5 5^2 = 194\,400$.

In total, there were 1932 applicants in that float. Due to the discreteness of attributes, the 1932 records provided 722 unique examples. These examples thus covered only $722/194\,400 = 3.7\%$ of the complete attribute space.

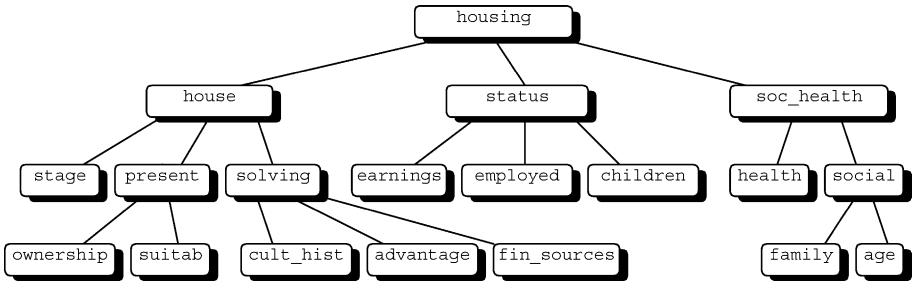

Fig. 3. Original model structure for housing loans allocation.

The primary goal of the experimental evaluation was to try to *reconstruct* the original decision model using the applicants' data to try to predict the observed loan priority decisions. The data was unstructured in the sense that it contained only applicants' input attributes and assigned priorities, but no intermediate concepts. Both unsupervised and supervised decompositions were carried out. The resulting models were interpreted, compared to the original one, and analyzed in terms of comprehensibility and classification accuracy. Finally, the ability of the model to generalize to unobserved combinations of attribute values was assessed by a cross-validation method.

3.1. Supervised decomposition

The particular methodology we have used for supervised decomposition is as follows. We used HINT to examine the data set and propose best candidates for partitions. The candidates were examined by an expert, who assessed the quality of partitions. The expert was instructed to select the partition that seemed most meaningful and comprehensible. The selected partition was then used for decomposition, and the experiment continued with the newly constructed data sets. The expert used for supervised decomposition was a member of the team that had developed the original decision models 3 years before the study described here. Interviews with the expert confirmed that he did not recall exact detail of the original models due to the passage of time, and we are confident that our outcomes are not solely due to the expert's ability to remember the original hierarchies. Before the session, only the names of the input attributes and their meaning were revealed to the expert. Notice that the purpose of this test was not to exactly reproduce the original model, but rather to see if supervised decomposition can outperform the unsupervised one, and, specifically, if the decomposition can propose a limited list of concepts within which are those highly meaningful to the expert.

In the first stage of the analysis, the attributes were tested for redundancy. The attributes cult_hist and fin_sources were found redundant and removed from the database. The reason for redundancy is that these two attributes affect the loan priority only under some very special circumstances, which did not occur in the database. For example, cult_hist applies only to renewing a house that has been declared a cultural or historical monument; and there were no such houses in that float.

The resulting set of examples was examined for decomposition. The set of all 12 attributes is shown in Fig. 4a. All possible partitions with bound sets of two or three attributes were examined. From these, according to the partition selection measure (column multiplicity v), HINT proposed only the best candidates with $v=3$. Among the 120 possible bound sets of three attributes, there were 11 bound sets that minimized column multiplicity:

suitab	advantage	employed
advantage	stage	employed
advantage	employed	health
advantage	employed	family
earnings	employed	children
earnings	employed	health
earnings	employed	family
earnings	children	health
employed	children	health
employed	health	family
employed	health	age

Among these, the domain expert felt that the underlined bound set is the most comprehensible, and captures the concept of an applicants' current status. The resulting concept structure is given in Fig. 4b.

The data for earnings, employed and children were replaced with a single element representing the appropriate value for the status concept. These 'new' housing examples consisted of eight data elements and were examined in a similar fashion. Of the 56 possible partitions of bound sets of three elements, three candidates were shown to the domain expert:

ownership	suitab	advantage
suitab	advantage	stage
health	family	age

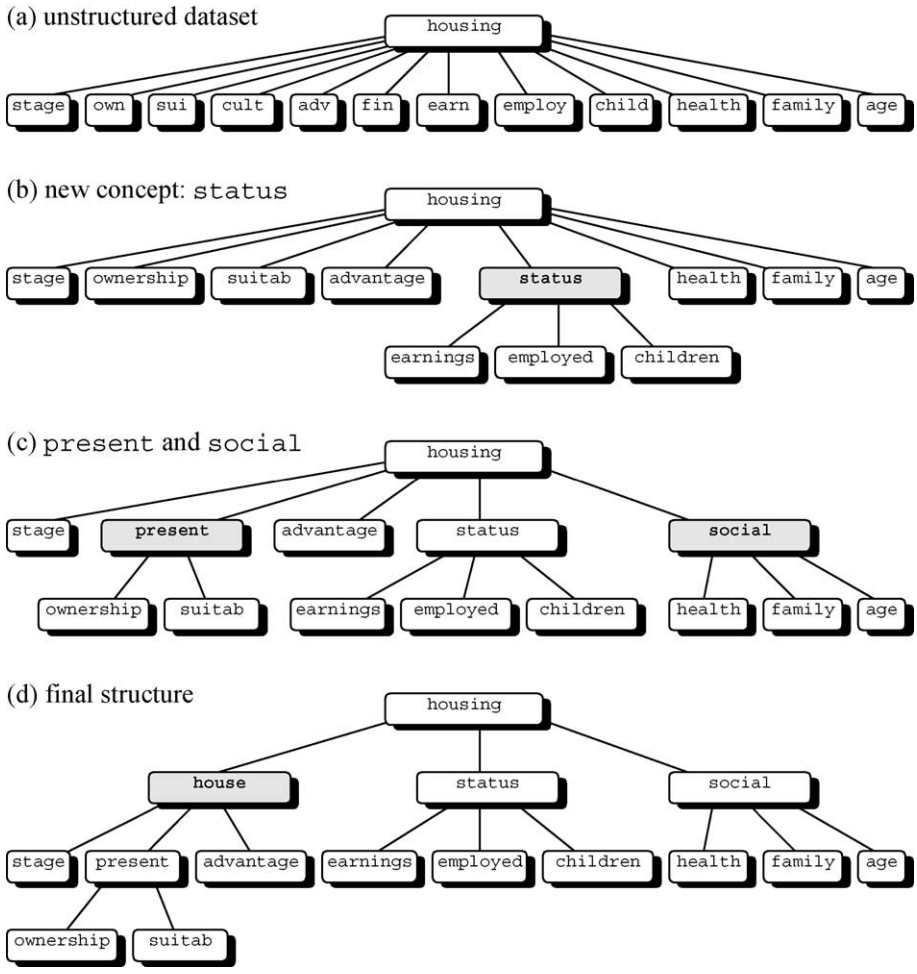

Fig. 4. Evolving concept structure for housing allocation decision model.

Again, the most favorable bound set is underlined, which was recognized as social and health condition of the applicant and formed a new 4-valued intermediate concept social.

The decomposition process continued in a similar fashion and resulted in two additional intermediate concepts: present (suitability of applicant's present housing, Fig. 4c) and house (overall housing conditions). This yielded an overall concept structure presented in Fig. 4d. Apart from the two excluded redundant attributes, the resulting concept structure is very similar to the structure actually used in the management decision support system to determine loan priority. Thus, when evaluated by the domain expert, the reconstruction was considered successful.

In each decomposition step, the selection of the most favorable bound set may not be straightforward. In our case, this was especially true for the first decomposition step, where the number of candidate partitions was quite high. The technique we employed was a gradual elimination of less favorable partitions. However, in the following decomposition steps, the number of candidate bound sets decreased substantially, which made the selection easier. The decreased number of candidates was due to the lower number of attributes and better coverage of the attribute space.

3.2. Unsupervised decomposition

To assess the benefit of user's interaction in the decomposition process, we used HINT in unsupervised mode and discovered the concept structure shown in Fig. 5. When the expert assessed this structure, he found that, in addition to subtrees that were identical or similar to the ones in the original model, some less intuitive intermediate concepts had been developed. For example, the unsupervised decomposition combined employed and health into c2, which was found difficult to interpret. In addition, as shown later in Section 3.5, the classification accuracy of this model was inferior to the model created by supervised decomposition. Therefore, the overall solution was not as satisfactory as the one obtained by supervised decomposition.

3.3. Interpretation of models

The sets of examples that describe the concepts in the resulting structure are considerably less complex than the initial one: while the initial set contains 722 examples, the most complex resulting set (housing) has only 38 examples, and all the remaining sets include less than 20 examples. In total, all the resulting sets include only 108 examples, which is a considerable reduction in comparison with the initial set. In addition, the decomposed sets use much less attributes.

Inspection of the resulting sets indicated that all were quite comprehensible. For example, even from the raw set it was easy to see that status depends monotonically on earnings, employed, and children. An even better interpretation was provided by a set of tools within DEX [2], which include decision rule induction methods and visualization tools [19]. The majority of the discovered concepts were viewed by the expert as representative of the relevant aspects of the decision context, and were consistent with the expert's expectations.

For a detailed example, let us look at the part of the hierarchy that combines the attributes ownership and suitab into the concept present. The attribute ownership represents the current ownership status of the loan applicant. It has five ordinal values that correspond to owning a flat on a permanent (1) or temporary (2) basis, renting a flat permanently (3) or temporarily (5), or living together with parents or other relatives (4). The second attribute, suitab, represents the suitability of applicant's current housing. It is measured relative to norms prescribed in the national housing law, using three values: (2) normal (within the norms), and (1) above and (3) below the norms. Notice that the values of both attributes are ordered so that higher values correspond to more difficult housing situations, which acquire a higher priority in the loan allocation procedure. These two attributes are combined into a concept called present, which assesses the applicant's present housing status in terms of three priority levels: (1) normal, (2) priority, and (3) high priority.

This part of the model is suitable for comparison because it occurs in both the original hand-crafted DEX model (Fig. 3) and in the model developed by HINT in supervised mode (Fig. 4d). The correspond-

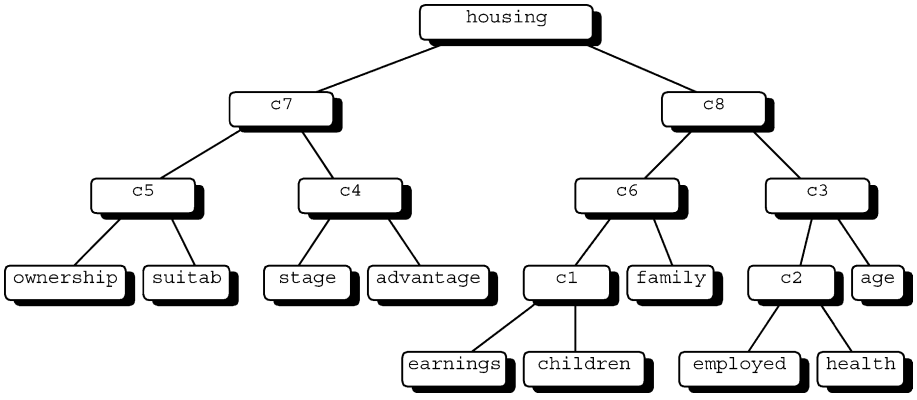

Fig. 5. Model structure developed by unsupervised decomposition.

ing two data sets are shown side by side in Table 3. There are 15 combinations of the values of ownership and suitab, which are mapped into the values of present in two different ways: as defined by the domain expert when developing the original DEX model, and as determined by HINT from data. Notice that HINT's values are just labels assigned in the decomposition process, which, nevertheless, correspond very well to the descriptive priority levels used in the DEX model.

The two data sets match in 12 of 15 rows, the three differences occurring in rows 2, 3, and 7. In rows 2 and 7, HINT did not assign any class at all. This is because there were no corresponding data items in the learning set, and no additional evidence was obtained by generalization (see Section 2.1). In row 3, there is an explicit difference between DEX and HINT, which assigned the classes 2 and 1, respectively. This difference occurs because the graph coloring algorithm (Section 2.1) identified two equally good labels: 1 and 2. The selection of 1 over 2 was arbitrary and was the direct result of the procedure's lack of knowledge of the original model. This was the only choice made arbitrarily in this data set. Notice that such situations occur where the data is insufficient to distinguish between the candidates. Function decomposition, as shown in Section 4.1, performs better when more data is available, as cases where HINT has to choose arbitrarily become less and less frequent.

Table 3
Original (DEX) and derived (HINT) data set for the concept present

	ownership	suitab	present	
			DEX	HINT
1	(1)owner	(1)above	(1)normal	1
2	(2)tmp – own	(1)above	(1)normal	none
3	(3)rent	(1)above	(2)priority	1
4	(4)relativ	(1)above	(2)priority	2
5	(5)tmp – rent	(1)above	(3)high_priority	3
6	(1)owner	(2)normal	(1)normal	1
7	(2)tmp – own	(2)normal	(1)normal	none
8	(3)rent	(2)normal	(2)priority	2
9	(4)relativ	(2)normal	(2)priority	2
10	(5)tmp – rent	(2)normal	(3)high_priority	3
11	(1)owner	(3)below	(2)priority	2
12	(2)tmp – own	(3)below	(3)high_priority	3
13	(3)rent	(3)below	(3)high_priority	3
14	(4)relativ	(3)below	(3)high_priority	3
15	(5)tmp – rent	(3)below	(3)high_priority	3

Overall, HINT fully reconstructed 12 of 15 data items (80%), and made only one improper class assignment. We believe this is a very good performance in such a difficult combinatorial task. This opinion was shared by the domain expert, who assessed the data set and thought it was comprehensible and relevant. He did not worry much about the differences between DEX and HINT; he explained that the two unassigned classes in rows 2 and 7 corresponded to quite rare and insignificant situations, and that row 3, although important, was difficult to classify anyway.

The data set constructed by HINT can be improved further by some additional inspection and interpretation. Namely, in its present form, HINT treats all variables as cardinal, that is, discrete and unordered. However, we do know that the attributes ownership and suitab, as well as the class present, are in fact ordered, and can take this into account to refine the HINT's data set in Table 3. For example, it is easy to see that the value in row 2 should have been 1 instead of "none". This is because in terms of ownership, row 2 lies just in between rows 1 and 3, while suitab is fixed. Since ownership is ordered and rows 1 and 3 both contain class 1, row 2 should contain the same class, too. Similarly, we can find out that the right class for row 7 is either 1 or 2, but definitely not 3, since this would contradict to row 8. Notice that this technique is available in DEX [2] and extensively used when developing new data sets. In this case, it has helped us to improve the data set developed by HINT.

3.4. Relation with MCDM

Another interesting way to look at data sets developed by DEX and/or HINT is to take the viewpoint of multi-criteria decision making (MCDM) [6]. There, the usual approach is to assess the relative weights of performance variables. To determine the overall utility of an option, we multiply the scores on the performance measures by the weights and sum them up. Performance variables and functions are typically defined on some continuous and preferentially ordered range, such as [0,1] or [0,100].

In this way, the HINT's data set (Table 3) can be interpreted as a utility function that maps the attributes ownership and suitab into present.

Fig. 6 shows that this is essentially a discrete function defined point by point for most combinations of the values of ownership and suitab. It increases monotonically in both directions and is symmetric for all the values of ownership that are greater than or equal to 3.

Notice that, strictly speaking, the function is defined only in the 13 points shown in Fig. 6, and the thick lines that connect the points serve only to aid visualization. This is different than most MCDM methods, where the functions are truly continuous and are typically defined over the full range of performance variables. Although the explicit point-by-point mapping as used in DEX and HINT is restricted to discrete attribute spaces, it does allow the ability to directly define relations between performance measures and provides a great deal of freedom to define non-linear relationships.

Despite the differences between MCDM and DEX/HINT, there are some approximate ways to bridge the two. DEX, for instance, uses a multiple linear regression technique to assess the weights of attributes from data sets [4]. It interprets the data items as points in a multi-dimensional space and approximates them by a linear function of the form $y = a_0 + a_1x_1 + \cdots + a_nx_n$. Here, $x_1, x_2, \ldots, x_n$ are input attributes and y is the class. To compensate for variables that have different number of discrete values, the input attributes are all uniformly mapped to the interval [0,1]. That is, for an ordinal variable v that can take the discrete values $v_1, v_2, \ldots, v_m$, these are transformed into:

$$x(v_i) = \frac{i-1}{m-1}, \qquad i = 1, 2, \ldots, m$$

The goal is to find the coefficients $a_0, a_1, a_2, \ldots, a_n$ so that the approximation is optimal in the least-squares sense. For better comprehensibility, the coefficients $a_1, a_2, \ldots, a_n$ are transformed into relative percentages and represented as weights $w_1, w_2, \ldots, w_n$:

$$w_i = \frac{100a_i}{(a_1 + a_2 + \cdots + a_n)}$$

For example, using the functions from Table 3, this method approximates the class present as follows:

$$\text{DEX}: \quad y = 0.83 + 1.60\frac{\text{ownership} - 1}{4} + 1.00\frac{\text{suitab} - 1}{2}$$

$$\text{HINT}: \quad y = 0.81 + 1.47\frac{\text{ownership} - 1}{4} + 1.16\frac{\text{suitab} - 1}{2}$$

For DEX, the weight of ownership is then estimated as $(100 \times 1.60)/(1.60 + 1.00) = 61.5\%$, and the

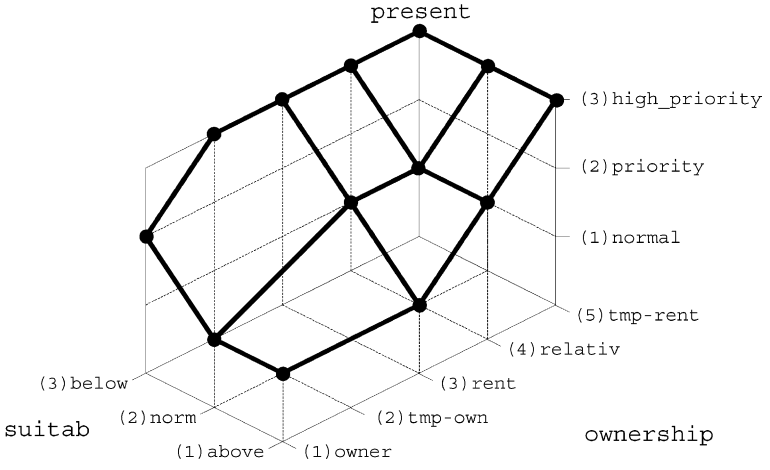

Fig. 6. HINT's data set for the class present interpreted as a utility function.

Table 4
Classification accuracies and their confidence intervals in percentage, determined by 10-fold cross validation

HINT (developed structure)	HINT (unsupervised)	C4.5
97.8 ± 1.8	94.7 ± 2.5	88.9 ± 3.9

weight of suitab as $(100 \times 1.00)/(1.60 + 1.00) = 38.5\%$. The corresponding weights for HINT are 55.9% and 44.1%. Thus, in both cases, ownership is substantially more important than suitab.

3.5. Cross-validation

The generalization quality of decomposition was assessed by 10-fold cross-validation. The initial set of examples was split to 10 subsets, and 10 experiments were performed using each individual subset as a test set and the examples in the remaining subsets as a single training set. HINT learned from the training sets and used either the concepts developed previously (Fig. 4d) or was used in the unsupervised mode on the training sets (i.e., 90% of original data). For comparison, we have also used C4.5, a machine learning program that induces decision trees from examples [18] and is considered a state-of-the-art generalization tool in machine learning. Notice, however, that C4.5 does not develop hierarchical decision models. Rather, it uses a different representation (classification trees) and does not explicitly develop new concepts.

The average classification accuracies and their confidence intervals obtained on the test sets (Table 4) clearly indicate that for this problem, the decomposition outperformed C4.5. It is further evident that the supervised method resulted in a classifier that was superior to that developed without user's interaction. However, it should be noted that the testing domain is, due to its original hierarchical structure, biased toward HINT.

4. Generalization, structure discovery and background knowledge

To additionally assess several properties of model discovery by function decomposition, we considered several multi-attribute decision models developed by experts who used DEX. From each model, the data was drawn such that each data instance encompassed some (random) combination of model's attributes and the corresponding classification. Notice that for the data sets constructed in this way, we have purposely excluded the values of internal concepts used in the original DEX hierarchies, thus removing any explicit knowledge about the concept hierarchy. The task for the function decomposition was to re-discover the original concept hierarchy. The advantage of such an approach is that we could vary the size of the data set from which HINT learned and observe how this affected the performance of function decomposition. Another, rather obvious advantage stems from the fact that hierarchical multi-attribute decision DEX models define both structure and intermediate concepts, so these were directly comparable to the models discovered by function decomposition.

Three DEX models, namely HOUSING, BANKING and BREAST, were used in our experiments. HOUSING is a housing loan allocation model introduced in the previous section [3]. BANKING was a model used for evaluation of business partners in banking [20], and BREAST a model for breast-cancer risk assessment [4]. Basic characteristics of these models are given in Table 5.

4.1. Generalization

We studied how the size of the training data (data from which we discover the model) affects HINT's ability to find a correct generalization. The generalization of HINT was assessed by a variant of stratified 10-fold cross-validation [15]. To obtain the learning curves (Fig. 7), p percentage of training examples in each iteration were randomly selected for learning, and the complete testing set was used to assess the classification accuracy; p ranged from 10% to 100%

Table 5
Basic characteristics of data sets: number of classes, number of attributes, average number of values per attribute, number of examples contained in the data set, and the proportion of examples belonging to the most frequent class

Data set	#class	#atts.	#val/att.	#examples	Majority class (%)
HOUSING	9	12	2.9	5000	29.9
BREAST	4	12	2.8	5000	41.5
BANKING	3	17	2.2	5000	40.8

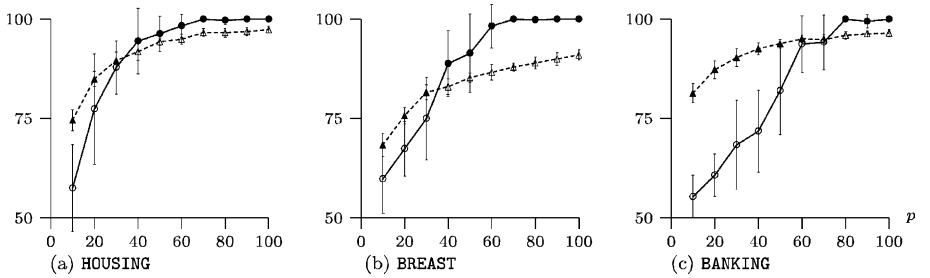

Fig. 7. Learning curves for unsupervised decomposition: classification accuracy as a function of training set size (○ and ● for HINT, △ and ▲ for C4.5).

in 10% steps. Again, HINT was compared to a machine learning tool C4.5, which is known for its good generalization. C4.5 was used with the default parameters except for `-m1` (classification tree leaves can contain a single example) and `-s` (tests can use attribute value subsetting), which were used so as to yield better classification accuracies. As suggested by Salzberg [23], a binomial test was used to test for significant differences between the methods using $\alpha = 0.01$. Drawing symbols used in Fig. 7 are ○ for HINT and △ for C4.5. Where the difference is significant, the symbol for the better classifier is filled (● for HINT and ▲ for C4.5).

Learning curves from Fig. 7 reveal that HINT generalizes well, but needs a substantial number of examples to obtain highly accurate decision models. There, with sufficient examples for learning, HINT's accuracy converged to 100%, which was never reached by C4.5.

Each decomposition step results in generalization thus increasing the coverage of the attribute space. To further investigate the generalization mechanism of function decomposition, we determined the portion of attribute space covered by the current classifier corresponding to the evolving concept hierarchy at each decomposition step. The results are shown for HOUSING and BREAST when learning from the complete data set of 5000 examples (Fig. 8). The generalization improves the most rapidly within the first few decomposition steps, where each decomposition step approximately doubles the coverage. The last few decomposition steps do not contribute much to the generalization, but rather further decompose the remaining larger data sets and thus reduce the complexity of the model.

4.2. Structure discovery

We assessed the convergence toward appropriate concept structures for DEX domains, depending on the size of training set, by using structure dissimilarity index (SDI, see Appendix A). SDI quantifies the dissimilarity between two concept structures, in our case between the ones discovered by HINT and developed using DEX. A lower SDI indicates a more similar structure, with an SDI of 0 indicating two identical structures.

For the three DEX models, the results using the same experimental setup as in Section 4.1 are given in Fig. 9. In all three cases, HINT converged to a single concept hierarchy that was very similar to the original DEX models. The primary differences involved HINT's tendency to decompose some of the more complex intermediate concepts of the original DEX models (an example is shown in Fig. 10). As with learning curves, the ability to discover relevant concept hierarchies highly depended on training set size.

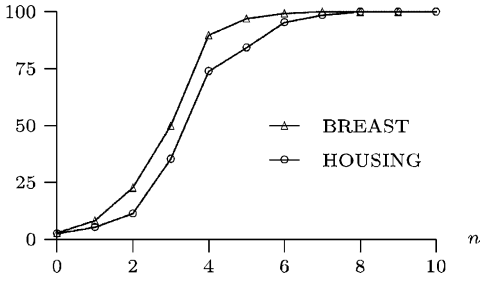

Fig. 8. Attribute space coverage in percentages as a function of the number of decomposition steps n.

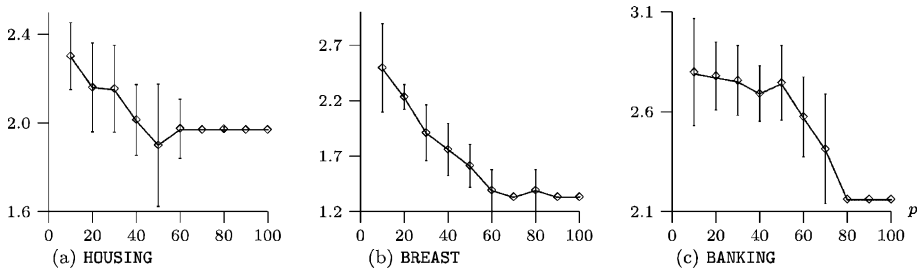

Fig. 9. Structure dissimilarity index as a function of training set size p.

4.3. Supervised decomposition

The high dependency on training set size and the observation that the main generalization occurs in the first decomposition steps indicate possible improvement by assisting HINT in early stages of decomposition. For the three DEX domains, we performed the same experiment as in Section 4.1 but in the first decomposition step enforced the creation of a single intermediate concept from the original DEX model. This concept can be regarded as a form of background knowledge, which could have been available had the expert assisted in the learning process. No assistance was then provided for subsequent decomposition steps. The results (Fig. 11) indicate a dramatic improvement in clas-

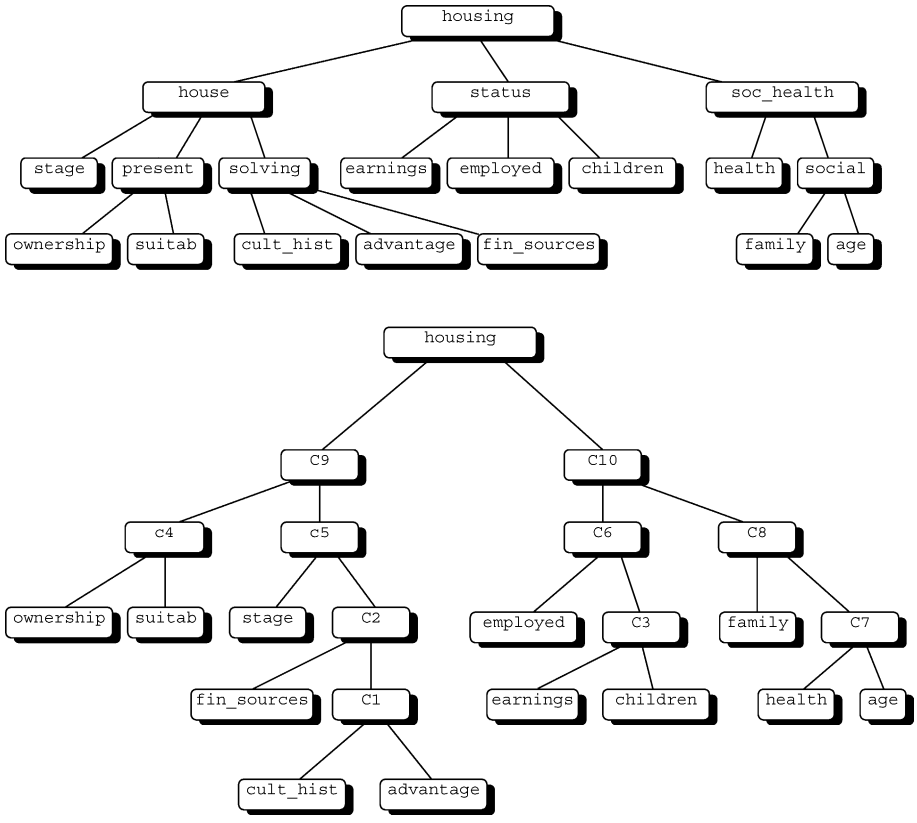

Fig. 10. The original (above) and decomposition-derived (below) structure for the HOUSING domain (SDI = 1.97).

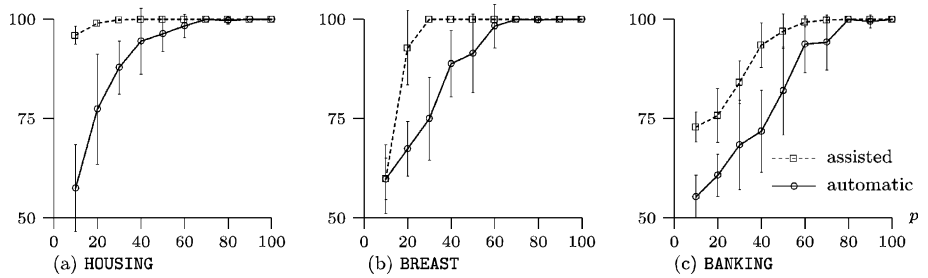

Fig. 11. Learning curves: assisted vs. automatic decomposition.

sification accuracy achieved with small training sets. This further confirms the hypothesis that the first few decomposition steps are crucial in obtaining an accurate decision model.

5. Related work

The problem of criteria identification and their structuring in terms of a decision model is central to multi-attribute decision making [11], decision analysis [6,17], group decision support [5], and related fields. The research reported here was motivated by a practical need for a method that would automate and/or assist the decision maker in developing a multi-attribute model from decision examples. The representation of decision models developed by the proposed method closely resembles that used in a multi-attribute decision support expert system shell DEX [2]. This approach differs from most of MCDM methods in two important ways: DEX's models use categorical and ordinal variables instead of continuous ones, and utility functions are defined point-by-point using data sets instead of employing some additive or multiplicative preference aggregation [6].

The decomposition method is based on the function decomposition approach to the design of digital circuits by Ashenhurst [1] and Curtis [7]. Their approach was advanced by Perkowski et al. [16], Luba [12], and Ross et al. [21]. Given a Boolean function partially specified by a truth table, their methods aim to derive switching circuits of low complexity. Function decomposition method was recently extended to handle multi-valued attributes and concepts and studied under the framework of machine learning [32]. While in this paper function decomposition is used without specific regard to noise handling, and consequently the resulting decision models are completely consistent with the data from which they were discovered, function decomposition methods that specifically deal with noise were proposed in Refs. [31,32].

The problem of developing hierarchical models from examples has been also studied within machine learning [26]. There, the decomposition approach was first used by Samuel [24] in checkers playing programs. His methods relied on a given concept structure but learned the corresponding functions from the training sets. Another technique that uses a predefined concept structure, known as structured induction [14], was independently developed by Shapiro [25] and originally used for the classification of a fairly complex chess endgame. It was shown that the obtained solutions were both comprehensible and of high classification accuracy.

The method presented in this paper is thus closely related to three primary research areas: it shares the motivation of multi-attribute decision making and structured induction, while the core of the method is based on Ashenhurst–Curtis function decomposition. In comparison with related work, this paper is original in the following aspects: adaptation of the function decomposition approach to the development of multi-attribute decision models, proposal and study of supervised decomposition, emphasis on generalization effects of decomposition, paying strong attention to the discovery of meaningful concepts, and experimental evaluation on a real-world decision problems.

6. Discussion

6.1. Lessons learned

We have assessed the applicability of the approach in a real-world housing loans allocation problem. It was demonstrated that the method was able to closely approximate the "right" decision model that was available for this problem. The reconstruction was carried out using 722 distinct decision examples taken from a real-life database of applicants. Although these examples sparsely covered the attribute space, the method succeeded in deriving a model of high comprehensibility and classification accuracy. The comparison of models developed by supervised and unsupervised decomposition revealed that human assistance had a positive effect on both the comprehensibility and classification accuracy. This example shows that the decomposition is a good generalization method and for this problem outperformed a state-of-the-art induction tool C4.5.

Further experiments on three other domains show that function decomposition may need a significant number of decision examples to obtain a model of high accuracy (see the learning curves in Fig. 7). While the number of examples required depends on the number of attributes, the number and distribution of utility values and overall nature of the domain, our experimental assessment indicates there is a heuristic method to determine if the set of decision examples is sufficient for developing an accurate model. Namely, the learning curves constructed from the data set as proposed in Section 4.1 should flatten out as the higher proportion of examples is presented to function decomposition. If this is not the case, and as indicated by learning curves from Fig. 10, then the involvement of domain expert in supervised decomposition becomes crucial.

With experiments in the reconstruction of the three hierarchical multi-attribute DEX decision models, we have gained further insight to some of the properties of function decomposition. First, we have observed that both HINT's generalization and structure discovery capabilities were strongly affected by the size of the data set from which it learned. However, given sufficient number of examples, HINT managed to discover models of good classification performance with the structure that closely matched the target (DEX's original) ones. Next, experiments revealed that first few decomposition steps may be the most influential ones: they contribute most to the generalization, and seem to be most prone to errors especially when small training sets are used.

Finally, and also consistently with the findings from the case study on housing loan-allocation database, we have learned that the use of additional expert's knowledge (background knowledge) in the form of partially known model structure may significantly improve HINT's performance. There, the "partially known model structure" was explicated as a user's preference over the set of candidates for new concepts. If some parts of model structure were indeed known explicitly, it could have been used again to guide decomposition and to construct the concepts present in the partially known model structure first, thus implicitly providing guidance over the selection of the concept candidates. Another, even more elaborate level of helping HINT to obtain accurate models is if some concepts (their attributes and corresponding decision table) are known in advance; the experiments in Section 4.3 show that knowing even a single such concept may significantly increase the accuracy of the resulting model.

6.2. Requirements and limitations of the approach

In any case, for a reasonably sized problem, function decomposition requires at least several hundred decision examples. Records of customer purchases, the actions of a web-site's users and routine medical decisions offer examples of situations where this approach can be utilized because of the storage of attributes of the decision scenario as well as the ultimate utility or value of the outcome.

The housing loan allocation example from Section 3 also points out an obvious problem related to data-driven model construction. In this case study, none of the loan applications included a house that would have a special historical value, so this attribute was not included in the resulting model. While this limits the approach in a rather obvious

way (one cannot model something that is not contained in the data), our method allows the user to add a missing attribute by manually editing the model in DEX after the initial hierarchy discovery.

In all our experiments, HINT was run with a limited number of attributes in the bound set, that is, in each decomposition step, only up to three attributes were considered to form a new concept. In general, this limit does not degrade HINT's performance in terms of the quality of the resulting model [31]. On the contrary, with small bound sets, more information is provided to the method on how to aggregate the values to form new concepts; partition matrices have more rows (more evidence) and less columns (smaller search space). Furthermore, notice that if larger bound sets would also be considered, the hierarchies developed may be the same or similar to the hierarchies developed with smaller bound sets, with a possible distinction that concepts closer to the root of the hierarchies may be discovered first. The only case where the limitation on the size of the bound set may have an effect is with cases where there are indeed indecomposable concepts that include many attributes. However, such concepts are not only rare but, due to the difficulty they provide for the explanation of decisions, deprecated in decision support.

6.3. Further work

The decomposition approach as presented in this paper is limited to consistent sets of examples using discrete attributes and utility. However, recently developed noise and uncertainty handling mechanisms [33], and an approach to handle continuously valued attributes [9] will enable HINT to be used in more general model developing tasks that are planned for the future.

Currently, HINT does not distinguish between cardinal and ordinal variables. As shown in Section 3.3, the models could improve when the ordering information is taken into account, so this would also be a welcome addition to HINT.

Another important issue is related to the interpretation of derived example sets. Currently, each concept in the developed decision model is defined by a data set, whose investigation and interpretation is left to the decision maker. Unless the set is particularly small, such an interpretation may be difficult, resulting in a less comprehensible decision model. Fortunately, there is a number of existing methods that could be used to assist in the interpretation. Two such techniques, visualization and regression, were sketched in Section 3.4, but there is a need for more methods, which could be adapted from the areas such as data analysis, data mining, and MCDM.

As indicated in Section 3.4, MCDM treats utility function quite differently to both DEX and HINT, but there are some approximate ways to convert between the two formalisms. This relationship should be explored in more depth, for instance: how to transform one representation into another, which transformations are admissible and under which conditions, and how to integrate the two approaches within a common modeling formalism.

7. Conclusion

We have demonstrated the utility of method called function decomposition for development of hierarchical multi-attribute decision models from the set of unstructured set of decision examples. The resulting model generalizes the decision examples and can serve for the evaluation of new alternatives. The method is implemented in a system called HINT.

The development of decision models can be carried out either autonomously or in the interaction with the decision maker. In the latter case, the method turns into a data-mining tool for data structuring and analysis: the decomposition assists in the identification of concepts, organizing them into a hierarchy and deriving the concept representations by example sets. In this process, the original data set is decomposed into a number of less complex data sets that are easier to interpret and analyze.

Function decomposition constructs decision models that are described through the formalism that is also used by a decision support tool DEX, a system that has been extensively used in practice to support decision making in areas such as economy, medicine, banking, strategic planning, and others. In other words, models constructed by HINT may be used in real-life decision support within DEX. This is a rather unique feature that makes HINT and the approach described in this paper a bridge between data mining

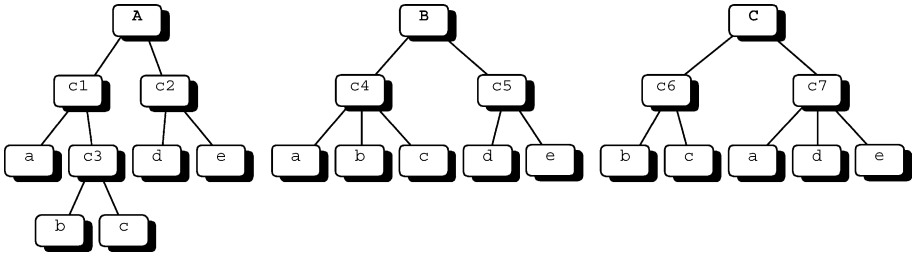

Fig. 12. Three example concept structures.

and multi-attribute decision support, further allowing the user to supervise and moderate the process of model development.

Acknowledgements

The work reported here was supported by the Slovenian Ministry of Education, Science and Sport, and by the EU project SolEuNet, IST-11495. The authors wish to thank Edvard Oven and Nevenka Fajdiga from the Housing Fund of the Republic of Slovenia for their dedication and expertise, and anonymous reviewers for their valuable comments and suggestions.

Appendix A. Structure dissimilarity index

Given a concept structure S and its two leaf nodes v_i and v_j, let their distance $d_s(v_i,v_j)$ be equal to the number of nodes one has to traverse to come from v_i to v_j. For example, for the structure A in Fig. 12, $d_A(a,c)=2$, $d_A(b,c)=1$, and $d_A(a,d)=3$.

Given two structures S_a and S_b with the sets of their leaf nodes V_a and V_b, respectively, their set of common leaf nodes is $V=V_a\cap V_b$. The structure dissimilarity index for structures S_a and S_b is then

$$\mathrm{SDI}(S_a,S_b)=\frac{1}{n(n-1)}\sum_{v_i\in V}\sum_{v_j\in V}\left|\, d_{S_a}(v_i,v_j) - d_{S_b}(v_i,v_j)\,\right|$$

where $n=|V|$. The higher the structure dissimilarity index, the less similar the two structures. Identical structures have SDI = 0.

For example, the pairwise dissimilarity index for structures A, B, and C from Fig. 12 are: SDI(A,B) = 0.6, SDI(A,C) = 1.0, and SDI(B,C) = 1.2. From these (and also by visual inspection) it may be concluded that the most similar structures of the three are A and B, and the least are B and C. Notice that while SDI has an intuitive meaning (average displacement of two nodes of the tree when compared to the reference hierarchy), the measure is not normalized and should not be used to compare the quality of discovered hierarchies.

References

[1] R.L. Ashenhurst, The Decomposition of Switching Functions, Technical report, Bell Laboratories BL-1 (11) (1952) 541–602.
[2] M. Bohanec, V. Rajkovič, DEX: an expert system shell for decision support, Sistemica 1 (1) (1990) 145–157.
[3] M. Bohanec, B. Cestnik, V. Rajkovič, A management decision support system for allocating housing loans, in: P. Humpreys, L. Bannon, A. McCosh, P. Migliarese (Eds.), Implementing System for Supporting Management Decisions, Chapman & Hall, London, 1996, pp. 34–43.
[4] M. Bohanec, B. Zupan, V. Rajkovič, Applications of qualitative multi-attribute decision models in health care, International Journal of Medical Informatics 58–59 (2000) 191–205.
[5] U. Bose, A.M. Davey, D.L. Olson, Multi-attribute utility methods in group decision making: past applications and potential for inclusion in GDSS, Omega 25 (6) (1997) 691–706.
[6] R.T. Clemen, Making Hard Decisions: An Introduction to Decision Analysis, Duxbury Press, Belmont, 1996.
[7] H.A. Curtis, A New Approach to the Design of Switching Functions, Van Nostrand, Princeton, 1962.
[8] Decision Analysis Society, MAU and AHP software, WWW page available at: http://www.faculty.fuqua.duke.edu/daweb/dasw1.htm (2002).
[9] J. Demšar, B. Zupan, M. Bohanec, I. Bratko, Constructing intermediate concepts by decomposition of real functions, in: M. van Someren, G. Widmer (Eds.), Machine Learning: ECML-97, Springer-Verlag, Berlin, 1997, pp. 93–107.

[10] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufman Publishers, San Francisco, 2001.
[11] R.L. Keeney, H. Raiffa, Decisions with Multiple Objectives: Preferences and Value Tradeoffs, Wiley, New York, 1976.
[12] T. Luba, Decomposition of multiple-valued functions, 25th Intl. Symposium on Multiple-Valued Logic (Bloomigton, IN), 1995, pp. 256–261.
[13] E.G. Mallach, Understanding Decision Support Systems and Expert Systems, Irwin, Boston, 1994.
[14] D. Michie, Problem decomposition and the learning of skills, in: N. Lavrač, S. Wrobel (Eds.), Machine Learning: ECML-95, Notes in Artificial Intelligence 912, Springer-Verlag, Berlin, 1995, pp. 17–31.
[15] D. Michie, D.J. Spiegelhalter, C.C. Taylor (Eds.), Machine Learning, Neural and Statistical Classification, Ellis Horwood, London,1994 [http://www.amsta.leeds.ac.uk/~charles/statlog/].
[16] A. Perkowski, T. Luba, S. Grygiel, P. Burkey, M. Burns, N. Iliev, M. Kolsteren, R. Lisanke, R. Malvi, Z. Wang, H. Wu, F. Yang, S. Zhou, J.S. Zhang, Unified Approach to Functional Decompositions of Switching Functions. Technical report, Warsaw University of Technology and Eindhoven University of Technology, 1995.
[17] L.D. Phillips, Decision analysis and its applications in industry, in: G. Mitra (Ed.), Computer Assisted Decision Making, North-Holland, Amsterdam, 1986, pp. 189–197.
[18] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufman Publishers, San Mateo, 1993.
[19] V. Rajkovič, M. Bohanec, Decision support by knowledge explanation, in: H.G. Sol, J. Vecsenyi (Eds.), Environments for Supporting Decision Processes, North Holland, Amsterdam, 1991, pp. 47–57.
[20] V. Rajkovič, E. Delidžakova-Drenik, B. Urh, Comparative analysis of three tools for the development and exploitation of expert system knowledge base (in Slovene), Informatica 15 (3) (1991) 22–28.
[21] T.D. Ross, M.J. Noviskey, D.A. Gadd, J.A. Goldman, Pattern theoretic feature extraction and constructive induction, Proc. ML-COLT '94 Workshop on Constructive Induction and Change of Representation (New Brunswick, NJ), 1994.
[22] T.L. Saaty, Multicriteria Decision Making: The Analytic Hierarchy Process, RWS Publications, Pittsburgh, 1993.
[23] S.L. Salzberg, On comparing classifiers: pitfalls to avoid and a recommended approach, Data Mining and Knowledge Discovery 1 (1997) 317–328.
[24] A. Samuel, Some studies in machine learning using the game of checkers II: recent progress, IBM Journal of Research and Development 11 (1967) 601–617.
[25] A.D. Shapiro, Structured Induction in Expert Systems, Turing Institute Press in association with Addison-Wesley Publishing Company, Wokingham, 1987.
[26] M.J. Shaw, Machine learning methods for intelligent decision support: an introduction, Decision Support Systems 10 (2) (1993) 79–83.
[27] M.J. Shaw, C. Subramaniam, G.W. Tan, M.E. Welge, Knowledge management and data mining for marketing, Decision Support Systems 31 (2001) 127–137.
[28] T.J. Stewart, A critical survey on the status of multiple criteria decision making theory and practice, Omega 20 (5/6) (1992) 569–586.
[29] W. Wan, M.A. Perkowski, A new approach to the decomposition of incompletely specified functions based on graph-coloring and local transformations and its application to FPGA mapping, Proceedings IEEE EURO-DAC-92, 1992, pp. 230–235. Hamburg.
[30] S.H. Zanakis, A. Solomon, N. Wishart, S. Dublish, Multi-attribute decision making: a simulation comparison of select methods, European Journal of Operational Research 107 (1998) 507–529.
[31] B. Zupan, Machine Learning Based on Function Decomposition, PhD Thesis, University of Ljubljana, Faculty of Computer and Information Sciences, 1997.
[32] B. Zupan, M. Bohanec, J. Demšar, I. Bratko, Learning by discovering concept hierarchies, Artificial Intelligence 109 (1999) 211–242.
[33] B. Zupan, I. Bratko, M. Bohanec, J. Demšar, Induction of concept hierarchies from noisy data, in: P. Langley (Ed.), Proc. International Conference on Machine Learning ICML-2000, Morgan Kaufman Publishers, San Francisco, 2000, pp. 1199–1206.

Marko Bohanec is a senior researcher at Jožef Stefan Institute, Department of Intelligent Systems, Ljubljana, and assistant professor in information systems at the School of Public Administration, University of Ljubljana. He obtained his MSc and PhD in Computer Science at the Faculty of Electrical Engineering and Computer Science, University of Ljubljana. His major research interests are related to decision support systems and data mining, and in particular to qualitative hierarchical modeling and machine learning. He has published in journals such as *Machine Learning*, *Information and Management*, *Artificial Intelligence*, *IEEE Intelligent Systems*, and *International Journal of Medical Informatics*.

Blaž Zupan received his MSc degree in Computer Science from the University of Houston, Texas, and PhD degree in Computer Science from the University of Ljubljana, Slovenia. He is an assistant professor at University of Ljubljana, Slovenia, and at Baylor College of Medicine, Houston, TX. His main research interest is in machine learning, data mining and systems for decision support. He applies these technologies to biomedical data analysis and construction of medical decision support systems. He is a principal author of HINT, a Hierarchy INduction Tool, a co-author of Orange, a component-based data mining freeware, and a co-author of GenePath, a system for genetic data analysis.