Learning Problem-Oriented Decision Structures from Decision Rules: The AQDT-2 System

Ryszard S. Michalski and Ibrahim F. Imam

Center for Artificial Intelligence
George Mason University
Fairfax, VA. 22030
{michalski, iimam}@aic.gmu.edu

ABSTRACT

A decision structure is an acyclic graph that specifies an order of tests to be applied to an object (or a situation) to arrive at a decision about that object. and serves as a simple and powerful tool for organizing a decision process. This paper proposes a methodology for learning decision structures that are oriented toward specific decision making situations. The methodology consists of two phases: 1—determining and storing declarative rules describing the decision process, 2—deriving on-line a decision structure from the rules. The first step is performed by an expert or by an AQ-based inductive learning program that learns decision rules from examples of decisions (AQ15 or AQ17). The second step transforms the decision rules to a decision structure that is most suitable for the given decision making situation. The system, AQDT-2, implementing the second step, has been applied to a problem in construction engineering. In the experiments, AQDT-2 outperformed all other programs applied to the same problem in terms of the accuracy and the simplicity of the generated decision structures.

Key words: machine learning, inductive learning, decision structures, decision rules, attribute selection.

1 Introduction

The main step in the development of an advisory system for decision making is the creation of a knowledge structure that characterizes the decision making process. A simple and effective tool for describing decision processes is a *decision structure*, which is an acyclic graph that specifies an order of tests to be applied to an object (or a situation) to arrive at a decision about that object. The nodes of the structure are assigned individual tests (which may correspond to a single attribute, a function of attributes, or a relation), the branches are assigned possible test outcomes (or ranges of outcomes), and the leaves are assigned one specific decision or a set of candidate decisions (with corresponding probabilities), or an *undetermined* decision. A decision structure reduces to a familiar decision tree, when each node is assigned a single attribute and has at most one parent, the branches from each node are assigned single values of that attribute, and leaves are assigned single, definite decisions. Thus, the problem of generating a decision structure is a generalization of the problem of generating a decision tree.

Decision trees are typically generated from a set of examples of decisions. The essential characteristic of any such method is the *attribute selection criterion* used for choosing attributes to be assigned to the nodes of the decision tree being built. Such criteria include the entropy reduction [12, 13], the gini index of diversity [4], and others (e.g., 5, 6, 11).

A decision tree/decision structure can be an effective tool for describing a decision process, as long as all the required tests can be measured, and the decision making situations it was designed for remain constant. Problems arise when these assumptions do not hold. For example, in some situations measuring certain attributes may be difficult or costly. In such situations it is desirable to reformulate the decision structure so that the “inexpensive” attributes are evaluated first (are assigned to the nodes close to the root), and the "expensive" attributes are evaluated only if necessary (are assigned to the nodes far away from the root). If certain attribute cannot be measured, it is useful to either modify the structure so that it does not contain that attribute, or—when impossible—to indicate alternative candidate decisions and their probabilities. A restructuring is also desirable, if there is a significant change in the frequency of occurrence of different decisions.

A restructuring of a decision structure (or a tree) in order to suit new requirements is usually quite difficult. This is because a decision structure is a form of procedural knowledge representation, which imposes an evaluation order of tests. In contrast, no evaluation order is imposed by a declarative representation, such as a set of decision rules. Tests (conditions) of rules can be evaluated in any order. Thus, for a given set of rules, one can usually build a large number of logically equivalent decision structures (trees), which differ in the test ordering. Due to the lack of “order constraints,” a declarative representation (rules) is much easier to modify to adapt to different situations than a procedural one (a decision structure or a tree). On the other hand, to apply decision rules to make a decision, one needs to decide in which order tests are evaluated, and thus, needs a decision structure.

An attractive solution of these opposite requirements is to acquire and store knowledge in a declarative form, and transform it to a decision structure when it is needed for decision making. This method allows one to create a decision structure that is most appropriate in a given decision making situation. Because the number of decision rules per decision class is usually small (much smaller than the number of training examples per class), generating a decision structure from decision rules can be potentially done much faster than generating it from training examples (methods generating decision trees from examples, are described, e.g., in 4, 12, 13, 14). Thus, this process could be done “on line,” without any delay noticeable to the user. Such “virtual” decision structures are easy to tailor to any given decision making situation.

This approach allows one to generate a decision structure that avoids evaluating an attribute that is difficult to measure or delay its evaluation. Initial ideas on this approach, and the first system implementing it, AQDT-1, have been described in [7]. This paper presents a new version of the system, called AQDT-2. The new system generates a goal-oriented decision structure from decision rules learned by either rule learning system AQ15 [10] or system AQ17, which has extensive constructive induction capabilities [2]. AQDT-2 has several new features, including: 1) a method for utilizing new attributes, not present in the original data, derived by constructive induction, 2) a method for controlling the degree of generalization needed during the development of the decision structure, 3) two new attribute selection criteria, 4) new method for combining different attribute selection criteria, 5) the ability to generate “unknown” nodes in situations, when there is insufficient information for generating a complete decision structure, 6) the ability to learn decision structures from "discriminant" decision rules, as well as "characteristic" rules, and 7) the ability to provide the most likely decision when the decision process stops, due to the inability to measure an attribute associated with some node. New features of AQDT-2 are demonstrated in an experiment on determining a decision structure for choosing wind bracings for tall buildings [1]. The results briefly illustrate how the system tailors decision structures to different decision making situations.

2 The AQDT-2 Method

This section describes the AQDT-2 method for building a decision structure from decision rules. The method builds a single-parent decision structure in a way similar to standard methods of building a decision tree from examples. The major difference is that it assigns tests (attributes) to the nodes using criteria based on the properties of the decision rules, rather than statistics characterizing the coverage of training examples. Other differences are that the branches may be assigned an internal disjunction of values (not only a single value as in decision trees), and leaves may be assigned a set of alternative decisions with probabilities. Tests are attributes or names standing for logical or mathematical expressions that involve several attributes or variables. In the following, we use the terms "test" and "attribute" interchangeably (to distinguish between an attribute and a name standing for an expression, the latter is called a *constructed* attribute). At each step, the method chooses a test (attribute) from an available set of tests by determining the *test utility* in the given set of decision rules. The test (attribute) utility is based on four elementary criteria: 1) *disjointness*, which captures the effectiveness of the test in discriminating among decision rules for different decision classes, 2) *importance,* which determines the importance of a test in the rules, 3) *value distribution*, which characterizes the distribution of the test importance over its of values, and 4) *dominance*, which measures the test presence in the rules. These criteria are defined below.

Cost : The cost of a test expresses the effort or cost needed to measure or apply the test.

Disjointness. The disjointness of a test is defined as the sum of the *class disjointness*—the disjointness of the test for each decision class. Suppose decision classes are C_1, C_2,..., C_m, and decision rulesets for these classes have been determined. Given test A, let V_1, V_2,....,V_m, denote sets of the values (outcomes) of A that are present in rulesets for classes C_1, C_2,..., C_m, respectively. If a ruleset for some class, say, C_t contains a rule that does not involve test A, than V_t is the set of all possible values of A (the domain of A).

*Definition 1***.** The *degree of class disjointness*, D(A, C_i) *of test A for the ruleset of class C_i*, is the sum of the degrees of disjointness, D(A, C_i, C_j), between the ruleset for C_i and rulesets for C_j, j=1, 2,...m, j ≠ i. The degree of disjointness between the ruleset for C_i and the ruleset for C_j is defined by:

$$D(A, C_i, C_j) = \begin{cases} 0, & \text{if } V_i \subseteq V_j \\ 1, & \text{if } V_i \supset V_j \\ 2, & \text{if } V_i \cap V_j \neq \phi \text{ or } V_i \text{ or } V_j \\ 3, & \text{if } V_i \cap V_j = \phi \end{cases} \qquad (1)$$

where ϕ denotes the empty set.

*Definition 2***.** The *disjointness* of the test A for evaluating a given set of decision rules is the sum of the degrees of class disjointness of each decision class:

$$\text{Disjointness}(A) = \sum_{i=1}^{m} D(A, C_i) \qquad \text{where } D(A, C_i) = \sum_{i=1,\, i\neq j}^{m} D(A, C_i, C_j) \qquad (2)$$

The disjointness of a test ranges from 0, when the test values in rulesets of different classes are all the same, to 3*m*(m-1), when every ruleset of a given class contains a different set of the test values. Selecting a test with the maximum possible disjointness produces a node of the decision structure whose children can be immediately assigned decision classes.

Importance. The second elementary criterion, the importance of a test, is based on the *importance score* (IS), introduced in [8]. In the obtained rules, each test is assigned a "score" that represents the total number of training examples, which are covered by the rules involving this test. Decision rules learned by an AQ learning program are accompanied with information on their strength. The rule strength is characterized by its *t-weight* and *u-weight.* The *t*-weight (*total-weight*) of a rule for some class is the number of examples of that class covered by the rule. The u-weight (*unique-weight*) of a rule for some class is the number of examples of that class covered only by this rule. The importance score of a test is the aggregation of the total-weights of all rules that contain that test in their condition part. Suppose given is a set of decision rules for m decision classes $C_1,..,C_m$, and there are n tests $A_1,..,A_n$ involved in these rules. The number of rules associated with class C_i is denoted by "r_i".

*Definition 3***.** The *importance score*, $IS(A_j)$, of the test A_j is determined by:

$$IS(Aj)= \sum_{i=1}^{m} IS(Aj, Ci) \qquad \text{where} \qquad IS(Aj, Ci.) = \sum_{k=1}^{r_i} Rik(Aj) \tag{3}$$

and R_{ik}, the weight of a test A_j in the rule R_k of class C_i is given by:

$$R_{ik}(A_j) = \begin{cases} t-\text{weight} & \text{if } A_j \text{ belongs to rule } R_{ik} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where i=1,..,n; ik=1,..,ri; j=1,..,m.

Value distribution. The third elementary criterion, value distribution, concerns the number of legal values of tests. Given two tests with the same importance score, it prefers the one with the smaller number of legal values. Experiments have shown that this criterion is especially useful when using discriminant decision rules.

*Definition 4***.** A *value distribution*, $VD(A_j)$ of a test A_j is defined by:

$$VD(A_j) = IS(A_j) / v_j \tag{5}$$

where "v" is the number of legal values of Aj.

Dominance. The fourth elementary criterion, dominance, prefers tests that appear in large number of rules, as this indicates their high relevance for discriminating among ruleset of given decision classes. Since some conditions in the rules have values linked by internal disjunction, counting such rules directly would not reflect properly their relevance. Therefore, for computing the dominance, the rules are counted as if they were converted to rules that do not have internal disjunction. Such a conversion is done by "multiplying out" the condition parts of the rules containing internal disjunction. For example, the condition part [x3=1 v 3]&[x4=1] is "multiplied out" to two rules with condition parts [x3=1]&[x4=1] and [x3=3]&[x4=1].

The above criteria can be combined into one general test ranking measure using the "lexicographic evaluation functional with tolerances" (LEF) [9]. LEF combines two or more elementary criteria by evaluating them one by one (in the order defined by LEF) on the given set of tests. A test passes to the next criterion only if it scores on the previous criterion within the range defined by the tolerance. The default order of the tests in LEF was chosen as:

<Cost $\tau 1$; Disjointness, $\tau 2$; Importance, $\tau 3$;Value distr., $\tau 4$; Dominance, $\tau 5$> (6)

where $\tau 1$, $\tau 2$, $\tau 3$ and $\tau 4$ are tolerance thresholds (in percentages); their default values are 0. The default value of the cost of each test is 1.

3 An Illustration of the Method

This section illustrates the method by applying it to a problem of learning a decision structure for determining the structural quality of a tall building design. The quality is classified into four classes: high (c1), medium (c2), low (c3), and infeasible (c4). In the first phase, program AQ15c (Michalski, et al. 1986; a new version written in "C" was used) determined decision rules from training examples. Each example was characterized by seven attributes: number of stories (x1), bay length (x2), wind intensity (x3), number of joints (x4), number of bays (x5), number of vertical trusses (x6), and number of horizontal trusses (x7). The data consisted of 335 examples, or which 220 (66%) were randomly selected to serve as training examples, and 115 (34%) were used for testing the obtained decision structures. Figure 1 shows decision rules obtained by AQ15c.

Decision class C1		
1	[x1=1][x6=1][x2=1,2][x3=1,2][x4=1,3][x5=1,2][x7=1..3]	(t :18, u :18)
2	[x1=3][x2=1][x3=1][x5=1][x6=1][x4=1,3][x7=1,3,4]	(t :3, u : 3)
3	[x1=5][x2=2][x3=2][x5=2][x4=3][x6=1][x7=2,3]	(t :2, u : 2)
4	[x1=1][x6=1][x2=2][x3=1,2][x4=3][x5=1,2][x7=4]	(t :2, u : 2)
5	[x1=3][x2=1][x4=1][x6=1][x7=1][x3=2][x5=1,2]	(t :2, u : 2)
6	[x1=1][x3=1][x6=1][x2=2][x4=1,3][x7=1,3][x5=3]	(t :2, u : 2)
7	[x1=2][x5=2][x2=1][x6=1][x3=1,2][x4=3][x7=4]	(t :2, u : 2)
Decision class C2		
1	[x1=2..4][x2=1,2][x3=1,2][x4=3][x5=2,3][x6=1][x7=2,3]	(t :28, u :19)
2	[x1=2..4][x2=2][x3=1,2][x4=3][x5=1,2][x6=1][x7=3,4]	(t :17, u : 6)
3	[x1=2,4][x2=1,2][x3=1,2][x4=3][x5=1][x6=1][x7=3,4]	(t :10, u : 4)
4	[x1=1,3,5][x2=1,2][x3=1,2][x4=3][x5=3][x6=1][x7=2,4]	(t :10, u : 2)
5	[x1=3,5][x2=1,2][x3=1,2][x4=3][x5=2,3][x6=1][x7=1,4]	(t : 9, u : 4)
6	[x1=2][x2=1,2][x3=1,2][x5=1,2,3][x4=1][x6=1][x7=1]	(t : 7, u : 6)
7	[x1=3,4][x2=2][x3=2][x4=1,3][x5=1,3][x6=1][x7=1,2]	(t : 6, u : 4)
8	[x1=3,5][x2=2][x3=1][x7=1][x4=1,2][x5=1,2,3][x6=1,3]	(t : 5, u : 5)
9	[x1=1][x2=1][x6=1][x3=1,2][x4=3][x5=1,2][x7=4]	(t : 4, u : 4)
10	[x1=1][x5=1][x2=2][x4=2][x6=2][x3=1,2][x7=1..3]	(t : 4, u : 4)
11	[x1=1,2][x2=1][x6=1][x3=1,2][x4=1,3][x5=3][x7=1,4]	(t : 4, u : 2)
Decision class C3		
1	[x1=2..5][x2=1,2][x3=1,2][x7=1..4][x4=1,2][x5=1,3][x6=2,4]	(t :41, u :32)
2	[x1=1..4][x2=1,2][x3=1,2][x4=2][x5=2][x6=2,3][x7=2..4]	(t :27, u :20)
3	[x1=1,3][x2=1][x3=1,2][x7=1..4][x4=2][x5=1,2][x6=2,3]	(t :19, u : 6)
4	[x1=1,2,4][x2=1,2][x3=1,2][x4=2][x5=2,3][x6=3,4][x7=1]	(t :13, u : 8)
5	[x1=5][x2=2][x4=2][x5=2][x3=1,2][x6=3][x7=2..4]	(t : 5, u : 5)
Decision class C4		
1	[x1=5][x2=2][x3=2][x4=1,3][x5=1][x6=1][x7=1..4]	(t : 4, u : 4)
2	[x1=5][x2=2][x3=1][x5=1][x6=1][x4=3][x7=3]	(t : 1, u : 1)

Figure 1: Decision rules determined by AQ15c from the wind bracing data.

Table 1 presents values of the elementary criteria for each attribute occurring in the rules, as used for determining the root of the decision structure. For each class, the row marked “Values” lists values occurring in the ruleset for this class. For evaluating the disjointness of an attribute, say **A**, each rule in the ruleset that does not contain attribute **A** is assume to contain an additional condition [**A**= a v b ...], where a, b, ... are all legal values of **A.**

Class		Attributes						
		x1	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**
C1	Values	1,2,3,5	1,2	1,2	1,3	1,2,3	1	1..4
	Class disjointness	1	1	0	2	1	3	0
C2	Values	1..5	1,2	1,2	1,2,3	1,2,3	1,2,3	1..4
	Class disjointness	2	1	0	3	1	4	0
C3	Values	1..5	1,2	1,2	1,2	1,2,3	2,3,4	1..4
	Class disjointness	2	1	0	4	1	8	0
C4	Values	5	1	1,2	1,3	1	1	1..4
	Class disjointness	0	0	0	2	0	3	0
Attribute Disjointness		5	3	0	11	3	18	3
Attribute Importance		245	82	25	245	233	245	181
Attribute value distribution		49	41	13	82	78	61	45
Attribute Dominance		45	34	42	33	40	30	54

Table 1: Values of selection criteria for each attribute for the wind bracing problem.

Assuming the default LEF, attribute x6 is chosen for the root (it has the highest disjointness). Branches stemming from the root are marked by single values or groups of values of x6, according to the way they occur in the decision rules (groups subsumed by other groups are removed (Imam and Michalski, 1993). The branches are assigned subsets of the rules containing these values. The process repeats for each branch until all rules assigned to each branch are of the same class. That class is assigned to the leaf.

Figure 2 presents a decision structure determined by AQDT-2 from decision rules shown in Figure 1 (using default LEF). The structure was tested on testing examples. The prediction accuracy was 88.7% (102 testing examples were classified correctly and 13 miss-classified). For comparison, program C4.5 for learning decision trees from examples was also applied to this same problem (Quinlan, 1990). The experiment was done with C4.5 using the default window setting (maximum of 20% the number of examples and twice the square root the number of examples), and set the number of trials to one. The decision tree learned by C4.5 had the prediction accuracy 84.3% (97 testing examples were classified correctly and 18 were miss-classified). The C4.5 tree was also considerably more complex (it had 17 nodes and 43 leaves).

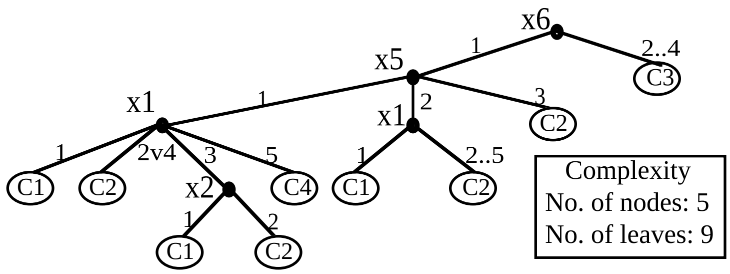


Figure 2: A decision structure determined by AQDT-2 for the wind bracing problem.

4 Discussion of the Special Aspects of the Method

4.1 Handling the Attribute Cost

As described in Sec. 2, the LEF criterion can take into consideration the cost of measuring tests (attributes). In the default LEF, the cost is the first criterion, and its tolerance is 0. If an attribute has high cost, or is impossible to measure (infinite cost), the LEF choses another, "cheaper" attribute, whenever possible. Figure 3 shows a decision structure obtained from the rules in Figure 1 under the condition that x1 is unavailable. Leaves marked ? denote situations in which a definite decision cannot be made without knowing x1. The decision tree was tested on 115 examples, of which 71 were classified correctly, 14 incorrectly, and 30 were assigned the "?" decision. Next subsection describes a method for determining a probability distribution for candidate decision classes instead of "?" decision..

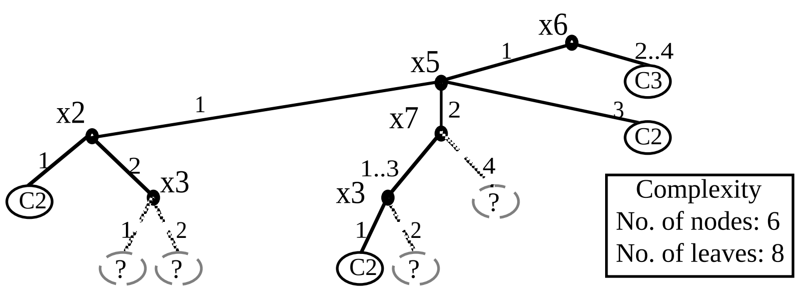


Figure 3: A decision structure learned without x1.

4.2 Assigning Decision Under Insufficient Information

When some attributes cannot be measured, the system may not be able to assign a definite decision. In the subsection above, such situations produce leaves denoted by "?". This symbol states that no definite decision can be made based on the available information, and signalizes the need for more information.

If no more information is available, but a decision must be made, it is useful to know the probabilities for different candidate decisions. The probabilities are determined on the basis of the class frequency at the given node. A frequency of class Ci at a node is calculated as the sum of the t-weighs of all the rules for that class that are assigned to the branch leading to that node. Figure 4 presents a decision structure from Figure 3 in which ? leaves were assigned candidate decisions with class probability estimates. For

the node X2, the frequences of the classes were: C1=31/45, c2=11/139 and C3=5/5. The estimate of the probability of C1 is then 31/47
under the node X2 was calculated

To illustrate this method, let us assume that we need to assign a decision to the example (x2=2; x3=2; x5=2; x6=1; x7=2) using or (x5=2; x6=1; x7=4). In such a case, there were no enough rules to cover such cases. AQDT-2 can generates a leaf node with all the possible decisions and provides approximate precision using the class frequency as in Figure 5.

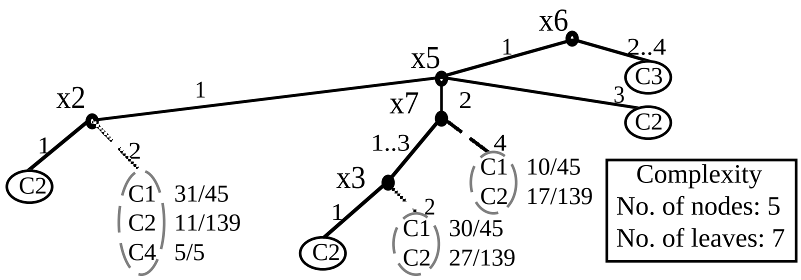
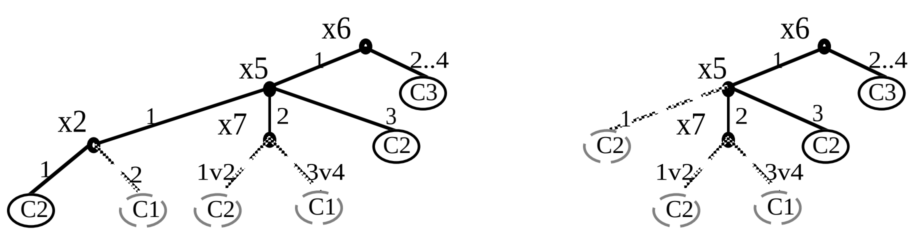


Figure 4: A decision structures with approximate decisions.

have the decision structure in Figure 3b, and we have the testing cases (x6=1; x5=2; x7=2), and (x6=1; x5=1; x4=2).

For the first example, Figure 4a shows a part of the decision structure that is most suitable for classifying the given example. Figure 4b provides a decision structure for solving both cases.



a)Partial decision for the first example b)Partial decision for both examples

Figure 4: A decision structures show the solution structure incompleteness.

A different method for handling the unavailability of an attribute was described by Quinlan [14]. His method gives a probabilistic class assignment, based on estimating the relative probability of a testing example belonging to different classes.

In AQDT-2, a probabilistic class assignment is used only when there is no alternative attribute to chose.

3.4 Decision Structure Pruning

Having noisy rules can affect the decision making process negatively. To prune the noisy rules a proposed method followed ideas introduced in earlier work [10, 7] where decision rules of small strength are pruned (e.g. rules which cover very few examples). The default setting of AQDT-2 prunes decision rules with strength of 3% or less of the total number of t-weight for the given decision class. Figure 6 shows a pruned decision structure learned by AQDT-2 after pruning rules with 10% or less t-weight, and it has a predictive accuracy of 88%.

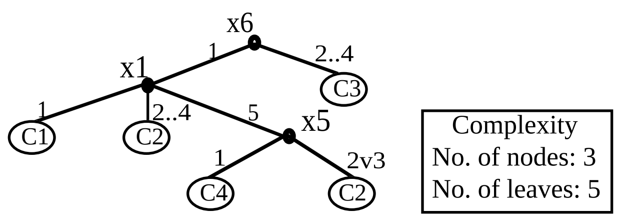


Figure 6: A pruned decision structure learned by AQDT-2.

3.5 Decision Structure Generalization

Seeking a general decision in some decision making situations is often desired, however it may influence the predictive accuracy [3]. AQDT-2 generalizes the decision structure, during the process of generating the decision structure, after selecting an attribute to be a node in the structure, and before splitting the rules into subsets each corresponds to one of its values. The class frequency is determined for each decision class, and the ratio between each class frequencies to the maximum class frequency at the given node is compared to a user-define threshold. If one or more ratios are greater than the threshold, the algorithm continues with generating branches for the given node. However, if there is no ratio greater than the defined threshold, a leaf node is generated and assigned to the class with maximum frequency. For example, the decision structure in Figure 1 is generated with 10% threshold. By increasing the threshold to 30%, the ratio of C1:C2 was less than 30% at the node (x6=1; x5=1; x1=3) which exchange the subtree at this point with a leaf node for C2, Figure 7a. Figure 7b shows a decision structure learned from the same rules with 45% generalization.

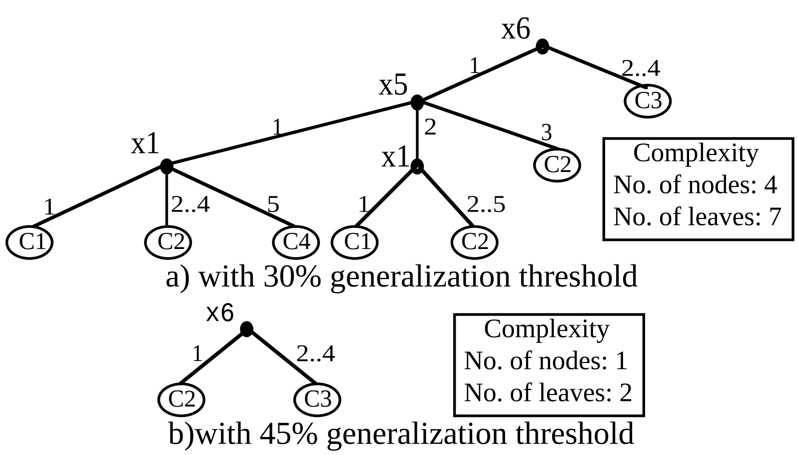


Figure 7: A decision tree learned form AQ15 rules with different generalization.

4. Conclusion

The system AQDT-2 determines problem-oriented decision structures from decision rules generated by an AQ-type inductive learning program. The system is quite efficient, because it is easier to generate a decision structure tailored to any given decision making situation from rules than to modify a decision structure once created [7]. The method uses an attribute ranking criterion composed of four elementary criteria: the disjointness, the importance, the value distribution, and the dominance of an attribute in the decision rules. AQDT-2 provides a set of new task-oriented features including: a method for controlling the degree of generalization needed during the development of the decision structure; two new attribute selection criteria; different methods for combining the attribute selection criteria; the ability to generate “unknown” nodes in situations when there is insufficient information for generating a complete decision structure; learning decision structures from "discriminant" rules, as well as "characteristic" rules; and the ability to provide the most likely decision when the decision process stops due to the inability to evaluate an attribute associated with an intermediate node. The new features of AQDT-2 are demonstrated in an experiment concerned with determining a decision structure for wind bracing design. The results shows how the system tailors decision structures to different decision making situations.

A major advantage of the proposed method is that it allows one to efficiently determine a decision structure that is optimized for any given decision making situation. For example, when some attribute is difficult to measure, the method creates a decision structure that shows in which situations measuring this attribute can be avoided. The method is quite efficient, and the time of determining a decision structure from decision rules in the cases we investigated was negligible. Therefore, it is easy to experiment with different criteria for structure generation in order to obtain the most desirable structure.

ACKNOWLEDGMENTS

The authors thank Eric Bloedorn for insightful comments on the earlier draft of the paper. This research was conducted in the Center for Artificial Intelligence at George Mason University. The Center’s research is supported in part by the Office of Naval Research under grant No. N00014-91-J-1351, in part by the National Science Foundation under grant No. IRI-9020266, in part by the Defense Advanced Research Projects Agency under the grant No. N00014-91-J-1854, administered by the Office of Naval Research, and the grant No. F49620-92-J-0549, administered by the Air Force Office of Scientific Research.

REFERENCES

1. **Arciszewski, T, Bloedorn, E., Michalski, R., Mustafa, M., and Wnek, J.,** "Constructive Induction in Structural Design", Report of Machine Learning and Inference Labratory, MLI-92-7, Center for AI, George Mason Un., 1992.

2. **Bloedorn, E., Wnek, J., Michalski, R.S., and Kaufman, K.**, "AQ17: A Multistrategy Learning System: The Method and User’s Guide”, Report of Machine Learning and Inference Labratory, MLI-93-12, Center for AI, George Mason Un. 1993.

3. **Bohanec, M. and Bratko, I.,** “Trading Accuracy for Simplicity in Decision Trees”, Machine Learning Journal, Vol. 15, No. 3, Kluwer Academic Publishers, 1994.

4. Breiman, L., Friedman, J.H., Olshen, R.A. & Stone, C.J., "Classification and Regression Structures", Belmont, California: Wadsworth Int. Group, 1984.

5. Cestnik, B. & Bratko, I., "On Estimating Probabilities in Structure Pruning", *Proceeding of EWSL 91*, (pp. 138-150) Porto, Portugal, March 6-8, 1991.

6. Cestnik, B. & Karalic, A., "The Estimation of Probabilities in Attribute Selection Measures for Decision Structure Induction" in *Proceeding of the European Summer School on Machine Learning*, July 22-31, Priory Corsendonk, Belgium, 1991.

7. Imam, I.F. and Michalski, R.S., "Learning Decision Structures from Decision Rules: A method and initial results from a comparative study", in *Journal of Intelligent Information Systems JIIS,* Vol. 2, No. 3, pp. 279-304, Kerschberg, L., Ras, Z., & Zemankova, M. (Eds.), Kluwer Academic Pub., MA, 1993.

8. Imam, I.F., Michalski, R.S. and Kerschberg, L., "Discovering Attribute Dependence in Databases by Integrating Symbolic Learning and Statistical Analysis Techniques", *Proceeding of the First International Workshop on Knowledge Discovery in Database*, Washington, D.C., July, 11-12, 1993.

9. Michalski, R.S., "AQVAL/1-Computer Implementation of a Variable-Valued Logic System VL1 and Examples of its Application to Pattern Recognition", *Proceeding of the First International Joint Conference on Pattern Recognition*, (pp. 3-17), Washington, DC, October 30- November 1, 1973.

10. Michalski, R.S., Mozetic, I., Hong, J. & Lavrac, N., "The Multi-Purpose Incremental Learning System AQ15 and Its Testing Application to Three Medical Domains", *Proceedings of AAAI-86*, (pp. 1041-1045), Philadelphia, PA, 1986.

11. Mingers, J., "An Empirical Comparison of selection Measures for Decision-Structure Induction", *Machine Learning*, Vol. 3, No. 3, (pp. 319-342), Kluwer Academic Publishers, 1989a.

12. Quinlan, J.R., "Discovering Rules By Induction from Large Collections of Examples", in D. Michie (Edr), *Expert Systems in the Microelectronic Age,* Edinburgh University Press, 1979.

13. Quinlan, J.R., "Learning efficient classification procedures and their application to chess end games" in R.S. Michalski, J.G. Carbonell and T.M. Mitchell, (Eds.), *Machine Learning: An Artificial Intelligence Approach*. Los Altos: Morgan Kaufmann, 1983.

14. Quinlan, J. R. "Probabilistic decision structures," in Y. Kodratoff and R.S. Michalski (Eds.), *Machine Learning: An Artificial Intelligence Approach, Vol. III*, San Mateo, CA, Morgan Kaufmann Publishers, (pp. 63-111), June, 1990.

1
AQDT-2 is concerned with the second step only
There is no comparison in this paper due to the space limitation
196 words, we want to reduce to arround 150

2
We need to explain why not to use the decision rules directly for decision making.
Answer: it is too difficult to adapt the decision rules for different decision making situations due to the independence among their conditions.
Also, it is most likely to ignore testing rules which for example contain costly attributes while they are the best rules for achieving the correct decision. On the other hand, classifying the decision rules by a decision structure increases the chances of having fair decision.