Transforming Rules and Trees into Comprehensible Knowledge Structures

Brian R. Gaines
Knowledge Science Institute
University of Calgary
Alberta, Canada T2N 1N4
gaines@cpsc.ucalgary.ca

Abstract

The problem of transforming the knowledge bases of expert systems using induced rules or decision trees into comprehensible knowledge structures is addressed. A knowledge structure is developed that generalizes and subsumes production rules, decision trees, and rules with exceptions. It gives rise to a natural complexity measure that allows them to be understood, analyzed and compared on a uniform basis. The structure is a directed acyclic graph with the semantics that nodes are premises, some of which have attached conclusions, and the arcs are inheritance links with disjunctive multiple inheritance. A detailed example is given of the generation of a range of such structures of equivalent performance for a simple problem, and the complexity measure of a particular structure is shown to relate to its perceived complexity. The simplest structures are generated by an algorithm that factors common sub-premises from the premises of rules. A more complex example of a chess dataset is used to show the value of this technique in generating comprehensible knowledge structures.

1 Introduction

It is fitting to commence this paper with a quotation from Ross Quinlan's foreword to the first volume of the Knowledge Discovery in Databases series:

> *"In one of several illuminating essays on different facets of knowledge, Donald Michie (1986) identifies* concept expressions *as those correct and effectively computable descriptions that can also be assimilated and used by a human being. As a counterexample, he cites a case in which ID3 derived a decision tree for a chess end game from a complete set of positions. The tree was absolutely correct and computationally efficient but, alas, completely incomprehensible to human chess experts. As he put it, "It was not a question of a few glimmers of sense here and there scattered through a large obscure structure, but just a total blackout." In Michie's view, which I share, such a structure does not qualify as knowledge." (Quinlan, 1991)*

The inductive models produced of significant databases typically have such large trees or numbers of rules that they are not comprehensible to people as meaningful knowledge from which they can gain insights into the basis of decision making. This problem is common to both the manually and inductively derived rule sets, and seems intrinsic to the use of tree or rule structures as the basis of performance systems (Li, 1991).

It is not obvious that an excellent performance system necessarily implies the existence of a comprehensible knowledge structure to be discovered. Human practical reasoning does not require overt knowledge structures (Gaines, 1993b), and the supposition of an implicit structure, or ‘mental model’, is a construct of the observer (Clancey, 1993). In the knowledge acquisition community ‘expertise transfer’ paradigms have been replaced in recent years by ‘knowledge modeling’ paradigms (Schreiber, Wielinga and Breuker, 1993) that impute the resultant overt knowledge to the modeling process, not to some hypothetical knowledge base within the expert. Clancey, who has played a major role in promoting this paradigm shift (Clancey, 1989; 1993), has done so in part based on his experience in developing overt knowledge structures from MYCIN rules to support GUIDON (Clancey, 1987) as a teaching system based on MYCIN.

The development of excellent performance systems will remain a major practical objective regardless of the comprehensibility of the basis of their performance. However, the challenge of increasing human knowledge through developing understanding of that basis remains a significant research issue in its own right. Can one take a complex knowledge base that is difficult to understand and derive from it a better and more comprehensible representation? If so, to what extent can the derivation process be automated?

This paper presents techniques for restructuring production rules and decision trees to generate more comprehensible knowledge structures.

2 Comprehensibility of Knowledge Structures

In attempting to improve the comprehensibility of knowledge structures it would be beneficial to have an operational and psychologically well-founded measure of comprehensibility. However, there are in general no such measures. This is not to say that one has to fall back on subjective judgment alone. There are general considerations that smaller structures tend to be more comprehensible, coherent structures more meaningful, those using familiar concepts more understandable, and so on. The internal analog weights and connections of neural networks with graded relations of varying significance are at one extreme of incomprehensibility. Compact sets of production rules are better but not very much so if there are no clear relations between premises or obvious bases of interaction between the rules. Taxonomic structures with inheritance relations between concepts, and concept definitions based on meaningful properties, are probably most assimilable by people, and tend to be the basis of the systematization of knowledge in much of the scientific literature. Ultimately, human judgment determines what is knowledge, but it is not a suitable criterion as a starting point for discovery since the most interesting discoveries are the ones that are surprising. The initial human judgment of what becomes accepted as an excellent knowledge structure may be negative. The process of assimilation and acceptance takes time.

One feature of knowledge structures, that is highly significant to discovery systems, is that *unicity*, the existence of one optimal or preferred structure, is the exception rather than the rule. There will be many possible structures with equivalent performance as models, and it is inappropriate to attempt to achieve unicity by an arbitrary technical criterion such as minimal size on some measure. The relative evaluation of different knowledge structures of equivalent performance involves complex criteria which are likely to vary from case to case. For example, in some situations concepts may be preferred that are deemed more appropriate in being usual, familiar, theoretically more interpretable or coherent, and so on. A structure based on these concepts may be preferred over a smaller one that uses less acceptable concepts. Thus, a discovery system that is able to present alternatives and accept diverse criteria for discrimination between them may be preferred over one that attempts to achieve unicity through purely combinatorial or statistical criteria. On the other hand, it is a significant research objective in its own right to attempt to discover technical criteria that have a close correspondence to human judgment.

These have been the considerations underlying the research presented in this paper: to generate knowledge structures that have a form similar to those preferred by people; to explore a variety of possible structures and accept as wide a range of possible of external criteria for assessing them automatically or manually; and to develop principled statistical criteria that correspond to such assessments to the extent that this is possible.

3 Exception Directed Acyclic Graphs (EDAGs)

The starting point for the research has been empirical induction tools that generate production rules or decision trees. One early conclusion was that a hybrid structure that had some of the characteristics of both rules and trees was preferable to either alone. This structure can be viewed either as a generalization of rules allowing exceptions, or as a generalization of trees such that the criteria at a node do not have to be based on a single property or be mutually exclusive, and the trees are generalized to partial orders allowing branches to come together again. These generalizations allow substantially smaller knowledge structures to be generated than either rules or trees alone, and overcome problems of trees in dealing with disjunctive concepts involving replication of sub-trees.

Figure 1 exemplifies this knowledge structure, termed an *exception directed acyclic graph* (EDAG). It may be read as a set of rules with exceptions or as a generalized decision tree. Its operational interpretation is:

- All paths through the graph are traced from each of its root nodes.
- For each node on a path, if the premise (if any) is true then the conclusion (if any) is noted for that path, replacing any previous conclusion noted for the path.
- A path is traced until a premise fails to hold, or a leaf node is reached.
- When a path has been traced, any conclusion noted is asserted.

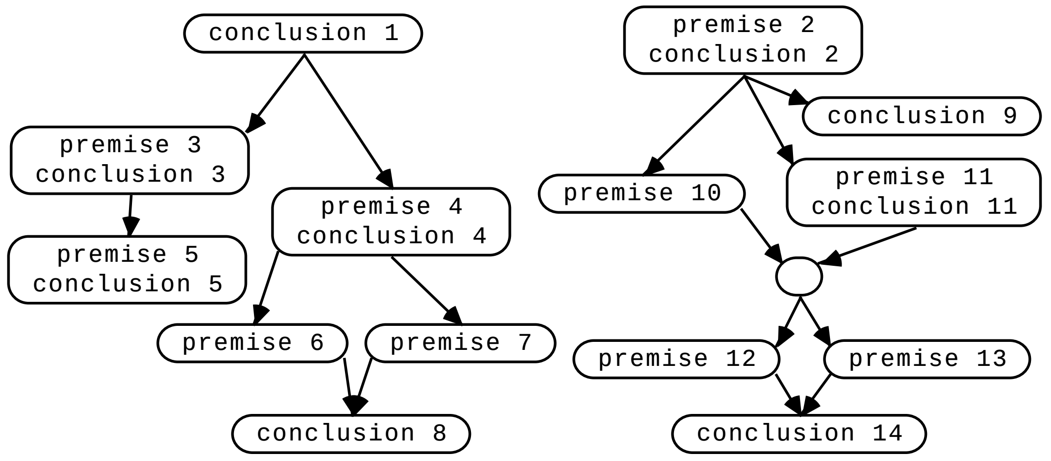


Figure 1 Knowledge structure—exception directed acyclic graph

Several features of EDAGs may be noted:

- The graph is not rooted and may have several disconnected parts.
- Concepts at a branch do not have to be mutually exclusive, so multiple conclusions can arise. An interesting example is conclusion 9 which is concluded unconditionally once premise 2 applies.
- As shown at premise 2, the structure is not binary. There may be more than two nodes at a branch.
- As shown at conclusion 8, the structure is not a tree. Branches may rejoin, corresponding to rules with a common exception or common conclusion.
- As shown at premise 6, a premise does not have to have a conclusion. It may just be a common factor to all the premises of its child nodes. As in decision trees, not all premises in an EDAG are directly premises of rules.
- As shown at conclusions 1 and 8, a conclusion does not have to have a premise. It may be a default conclusion, or a common conclusion to all the premises on its parent nodes.
- As shown by the null node between premises 10 and 12, it may be appropriate to insert an empty node to avoid arrows crossing.
- The notion of "premise" used is that of any potentially decidable predicate; that is, one with truth values, true, false or unknown. The unknown case is significant because, as illustrated later, conclusions from the EDAG as a whole may be partially or wholly decidable even though some premises are undecidable in a particular case—usually because some information is missing for the case.

A set of production rules without a default and without exceptions forms a trivial EDAG with no connections. An ID3-style decision tree forms an EDAG that is restricted to be a tree, with the set of premises fanning out from a node restricted to be tests of all the values of a particular attribute. Rules with exceptions form an EDAG in which every node has a conclusion. Rules with exceptions with their premises factored for common sub-premises (Gaines, 1991d) form a general EDAG.

The direction of arrows in Figure 1 indicates the decision-making paths. However, if the arrows are reversed they become the 'isa' arrows of a semantic network showing inheritance among premises, with multiple inheritance read disjunctively. For example the complete premise leading to the conclusion 14 is premise 2 AND (premise 10 OR premise 11) AND (premise 12 OR premise 13).

The complete premise for nodes with child nodes involves the negation of the disjunction of the child nodes, but not of their children. For example, the complete premise leading to the conclusion 1 is NOT (premise 3 OR premise 4), with premises 5 through 7 playing no role.

This last result is important to the comprehensibility of an EDAG. The 'meaning' of a node can be determined in terms of its path(s) back to the root and its child nodes. The other parts of the tree are irrelevant to whether the conclusion at that node will be selected. They remain potentially relevant to the overall conclusion in that they may contribute additional conclusions if the structure is not such that conclusions are mutually exclusive.

From a psychological perspective, the interpretation of the EDAG may be seen as each node providing a well-defined ‘context’ for its conclusion, where it acts as an exception to nodes back to the root above it, and it has as exceptions the nodes immediately below it. This notion of context is that suggested by Compton and Jansen (1990) in their analysis of ‘ripple-down rules’, the difference being that EDAG contexts are not necessarily mutually exclusive.

4 Inducing EDAGs

Induct (Gaines, 1989) is used to generate EDAGs through a three-stage process. First, it is used to generate rules with exceptions. Second, the premises of the rules are factored to extract common sub-premises (which are not necessarily unique, e.g. A AND B, B AND C and C AND A may be factored in pairs by A, B or C). Third, common exception structures are merged. Currently, a greedy algorithm is used that reports the first factorization found, and does not search over all factorizations to achieve a global minimum. The second and third stages of factoring can be applied to any set of production rules represented as an EDAG, and it is appropriate to do so to C4.5 and PRISM rules when comparing methodologies and illustrating EDAGs as knowledge structures.

The familiar contact lens prescription problem (Cendrowska, 1987) will be used to illustrate the EDAG as a knowledge structure. The problem involves three 2-valued attributes, one 3-valued attribute, and a 3-valued conclusion. Figure 2 shows the ID3 decision tree for the lens problem as an EDAG, and Figure 3 shows the PRISM production rules as an EDAG. The representation in itself adds nothing to the results, but the examples illustrate the subsumption of these two common knowledge structures. The representation of the rules can be improved by factoring the premises to produce the EDAG shown in Figure 4.

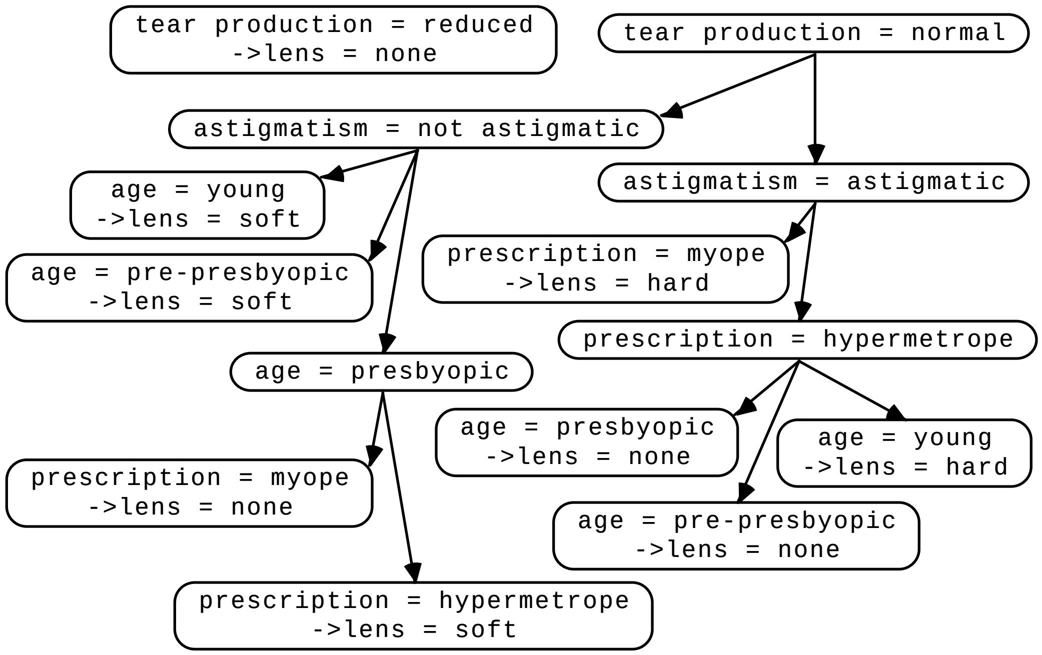


Figure 2 ID3 contact lens decision tree as EDAG

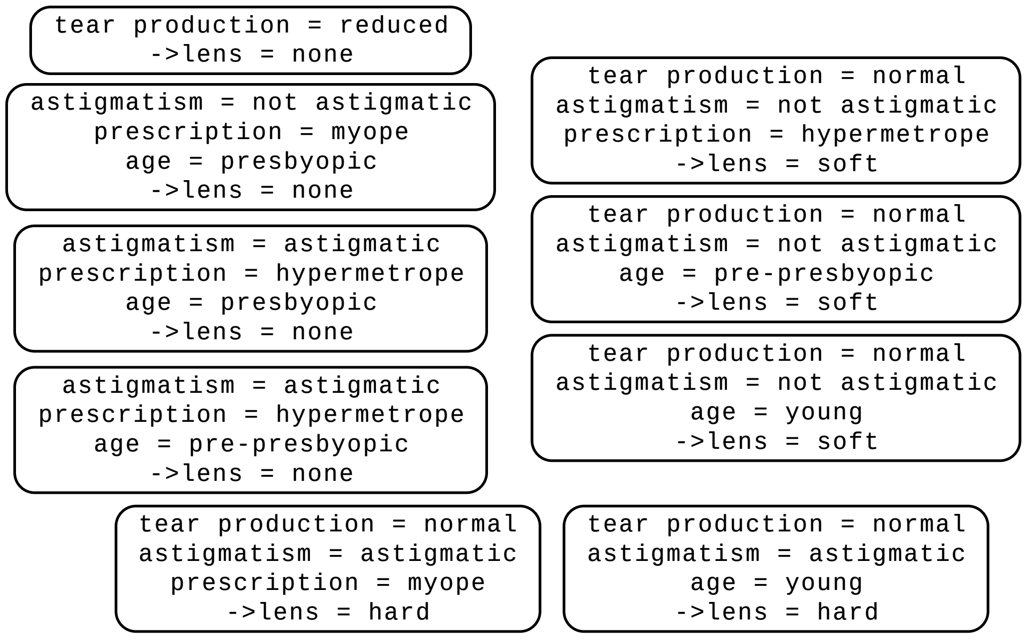

tear production = reduced ->lens = none
astigmatism = not astigmatic prescription = myope age = presbyopic ->lens = none
astigmatism = astigmatic prescription = hypermetrope age = presbyopic ->lens = none
astigmatism = astigmatic prescription = hypermetrope age = pre-presbyopic ->lens = none
tear production = normal astigmatism = not astigmatic prescription = hypermetrope ->lens = soft
tear production = normal astigmatism = not astigmatic age = pre-presbyopic ->lens = soft
tear production = normal astigmatism = not astigmatic age = young ->lens = soft
tear production = normal astigmatism = astigmatic prescription = myope ->lens = hard
tear production = normal astigmatism = astigmatic age = young ->lens = hard

Figure 3 PRISM contact lens production rules as EDAG

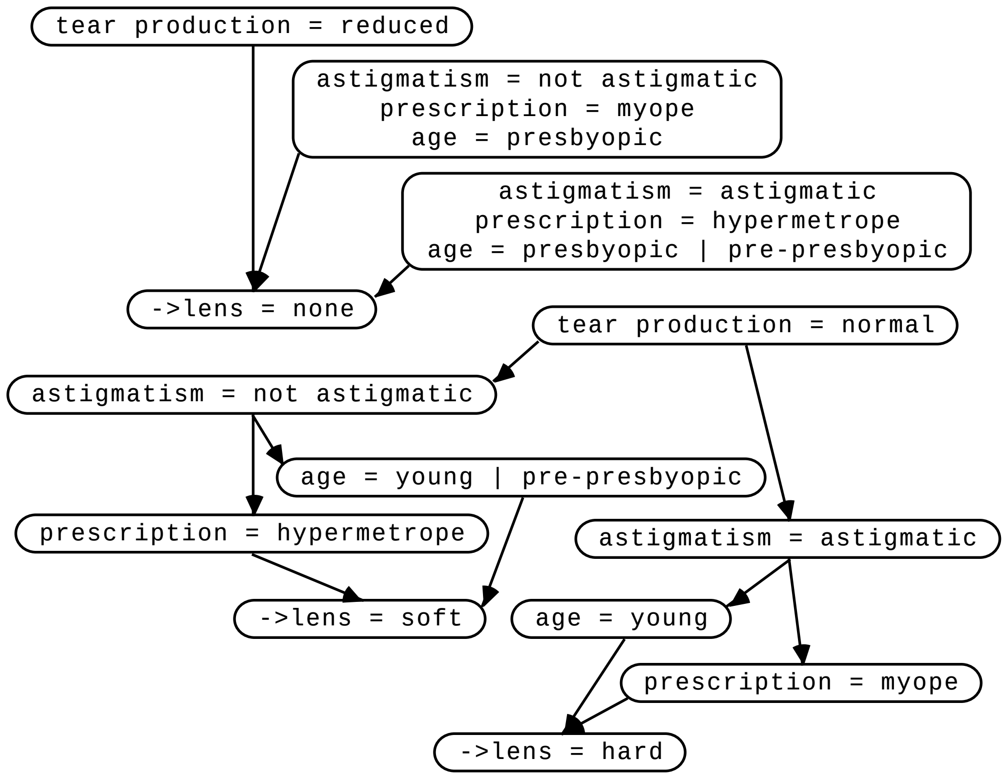

tear production = reduced
astigmatism = not astigmatic prescription = myope age = presbyopic
astigmatism = astigmatic prescription = hypermetrope age = presbyopic | pre-presbyopic
->lens = none
tear production = normal
astigmatism = not astigmatic
age = young | pre-presbyopic
prescription = hypermetrope
astigmatism = astigmatic
->lens = soft
age = young
prescription = myope
->lens = hard

Figure 4 PRISM contact lens production rules, factored as EDAG

Figures 2 through 4 are trivial EDAGs in the sense in that no exceptions are involved. More interesting examples can be generated using C4.5's methodology of: reducing the number of production rules by specifying a default conclusion; removing the rules with that conclusion; and making the other rules exceptions to the default. This can be applied to the PRISM rules in

Figure 4 by making “none” the default and removing the four rules with “none” as a conclusion on the left. Both PRISM and C4.5 then generate the same set of rules with exceptions which can be factored into the EDAG shown in Figure 5.

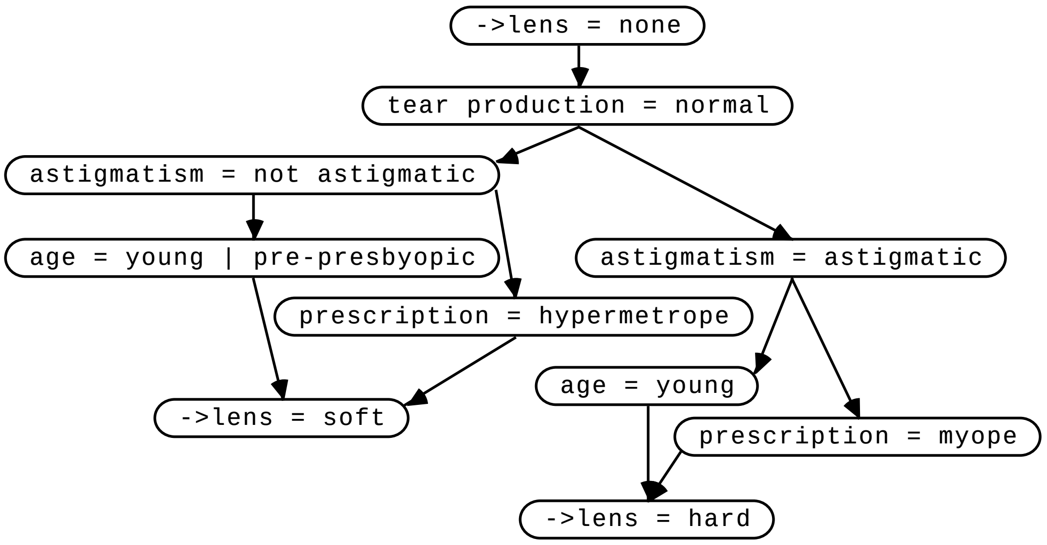


Figure 5 PRISM/C4.5 contact lens rules with default, factored as EDAG

Induct generates multi-level exceptions in that an exception may itself have exceptions. Figure 6 shows the EDAG generated by Induct from the 24 lens cases. On the left, the “soft” conclusion is generated by a simple premise that has an exception. On the right, the “hard” conclusion is generated by the same rules as in Figure 5.

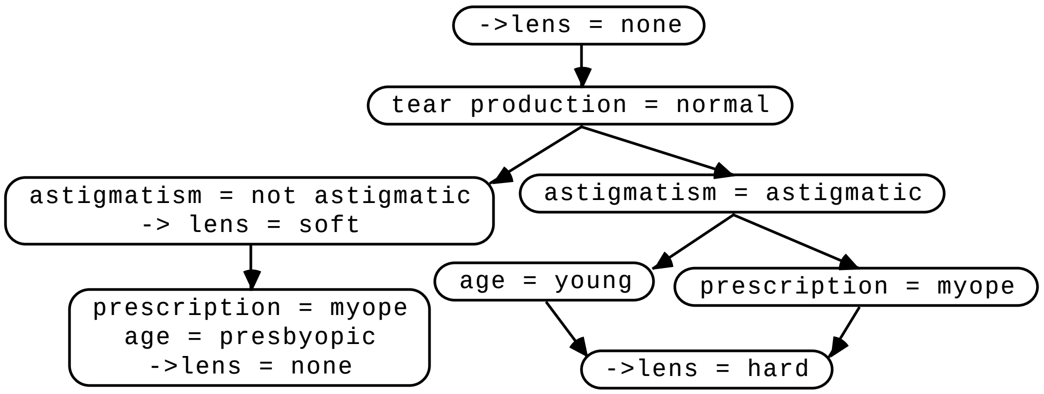


Figure 6 Induct rules with exceptions, factored as EDAG

Varying the distribution of the data set can lead to different exception structures. Figure 7 shows the EDAG is generated by Induct when the data is biased to have a lower proportion of the “hard” exceptions. 46 cases have been used—2 x 24 with the 2 examples of the “hard” exception removed from the second set. The data is logically unchanged, but the statistical test that Induct uses to determine whether to generate an exception is affected. The exceptions are now statistically more “exceptional”.

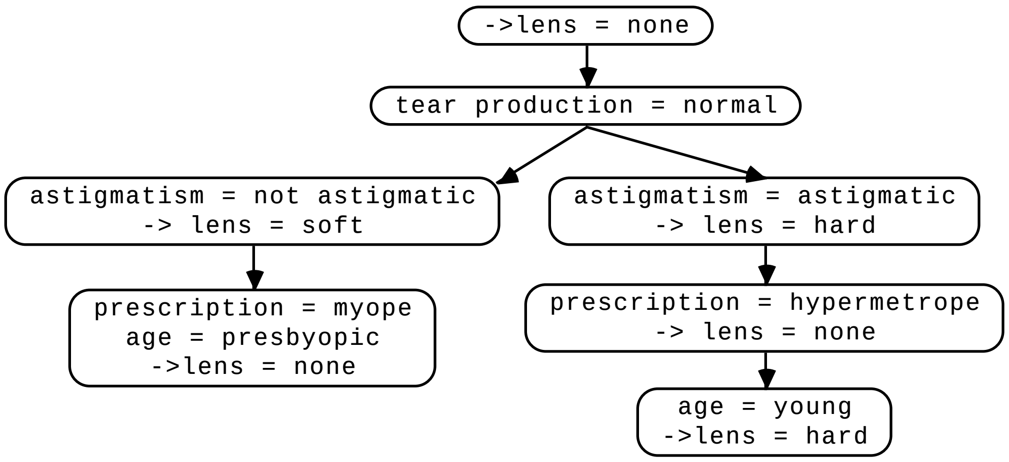

->lens = none
tear production = normal
astigmatism = not astigmatic -> lens = soft
astigmatism = astigmatic -> lens = hard
prescription = myope age = presbyopic ->lens = none
prescription = hypermetrope -> lens = none
age = young ->lens = hard

Figure 7 Alternative Induct rules with exceptions, factored as EDAG

5 Complexity Measures for EDAGs

Figures 2 through 7 show that many different EDAGs can be derived having the same performance. It is also apparent that the simpler ones present a more comprehensible knowledge structure. Figures 5 through 7 show that simple comprehensible knowledge structures are not unique. Each represents a reasonable basis for understanding and explaining contact lens decisions. Figure 6 may appear marginally better than the alternatives in Figures 5 and 7, but one can imagine a debate between experts as to which is best—Figure 5 has only one layer of exceptions—Figure 7 is more balanced in its treatment of astigmatism.

It is interesting to derive a complexity measure for an EDAG that corresponds as closely as possible to the psychological factors leading to judgments of the relative complexity of different EDAGs. As usual, the basis for a structural complexity measure is enumerative, in that one counts the components in the representation (Gaines, 1977). The obvious components are the nodes, the arcs, and the premise and conclusion clauses. Since the graphs are not trees, branches can rejoin with possible line crossings causing visual confusion. This correlates with the extent to which nodes are disjunctive and have several outgoing arcs. A possible measure is the number of outgoing arcs in excess of one for a node, which may be calculated as arcs + final nodes - nodes. Induct also allows the option of a disjunction of values within a clause, and such a disjunction is weighted by the number of values mentioned, e.g. x=5 counts 1, x BELONGS-TO {5, 8} counts 2, and x BELONGS-TO {5, [8, 12]} counts 3 (where [8,12] specifies the interval from 8 through 12).

The derivation of an overall complexity measure from these component counts has to take into account the required graph reductions which should lead to complexity measure reductions. The basic requirements are shown in Figure 8. It is apparent that clause and crossing reduction should dominate node reduction.

One suitable formula is that shown in Figure 8, that the:

complexity of an EDAG = (nodes + excess arcs x 2 + clauses x 2)/5 (1)

where the scaling by 5 is for normalization.

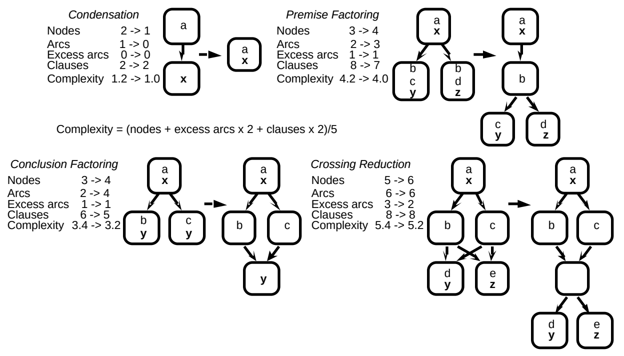


Figure 8 Complexity reductions required to correspond to graph reductions

For binary decision trees and production rules there are relations between the numbers in formula (1) that allow one to give the complexity measure more specific interpretations.:

complexity of a binary decision tree = number of nodes (2)

which is the usual measure of the complexity of a binary decision tree;

complexity of production rules = 0.6 x number of rules + 0.4 x number of premise clauses (3)

which is a weighted mean of two usual measures of the complexity of a set of production rules.

It would be appropriate to conduct empirical psychological experiments with a range of EDAGs and evaluate the correlation between the understanding of knowledge measured in various ways and the complexity measure developed here. Sufficient data would also allow other complexity measures to be evaluated. Nosek and Roth's (1990) empirical comparison of formal knowledge representation schema exemplifies what can be done. Shum and Hammond (1994) have surveyed studies of comprehensibility of data structures more generally.

Figure 9 shows the component counts and the computed complexity measures for the solutions to the lens problems shown in Figures 2 through 7. The computed complexity measure does appear to have a reasonable correspondence to the variations in complexity that are subjectively apparent.

Fig.	Example	Nodes N	Final Nodes F	Arcs A	Excess E = A +F-N	Clauses C	Complexity (N+2E+2C) /5
2	ID3	14	9	12	7	23	14.8
3	PRISM	9	9	0	0	34	15.4
4	PRISM, factored	13	3	13	3	19	11.4
5	C4.5 default, factored	10	2	11	3	11	7.6
6	Induct, factored	8	2	8	2	11	6.8
7	Induct variant, factored	7	2	6	1	13	7.0

Figure 9 Comparison of complexity of contact lens EDAGs

6 A Chess Example

The contact lens example is useful for illustration but too simple for the significance of the complexity reduction to be evaluated. This section presents a somewhat more complex example based on one of Quinlan’s (1979) chess datasets that Cendrowska (1987) also analyzed with Prism. The data consists of 647 cases of a rook versus knight end game situation described in terms of four 3-valued and three 2-valued attributes leading to one of two conclusions. All the solutions described are 100% correct.

Figure 10 shows the 30 node decision tree produced by ID3 and Figure 11 shows the 15 rules produced by C4.5 for the chess data. Cendrowska (1987) reports a substantially larger tree and the same number of rules with some additional clauses. Figures 12 and 13 shows the rules produced by C4.5 factored in 2 ways. The EDAG in Figure 13 introduces an additional (empty) node to make it clear that the possible exceptions are the same on both branches. Figure 14 shows an EDAG produced by Induct. All three EDAGs are interesting in not being trees and involving different simple presentations of the same knowledge.

Figure 15 shows the complexities of various EDAGs solving the chess problem. The decision tree published by Cendrowska (1987) is more complex than that produced by ID3. PRISM generates the same 15 production rules as does C4.5 except that two of the rules have redundant clauses. This has a small effect on the relative complexity of the production rules, but a larger effect on that of the factored rules since those from PRISM do not factor as well because of the redundant clauses. The three solutions shown in Figures 12 through 14 are similar in complexity as one might expect. They are different, yet equally valid, ways of representing the solution.

The reduction between the tree in Figure 10 or the rules in Figure 11 and the EDAGs in Figures 12 through 14 does seem to go some way to meeting Michie’s objections to trees or rule sets as knowledge structures that Quinlan (1991) cites. It is more plausible to imagine a chess expert using the decision procedures of Figures 12 through 14 than those of Figures 10 or 11.

```
line = f: game = safe (324.0)
line = t:                               (black king, rook, knight, in line)
|   r>>k = f: game = safe (162.0)
|   r>>k = t:                                     (rook bears on black king)
|   |   r>>n = f: game = safe (81.0)
|   |   r>>n = t:                                     (rook bears on knight)
|   |   |   k-n = 1:
|   |   |   |   n-wk = 2: game = safe (9.0)       (knight to white king = 2)
|   |   |   |   n-wk = 3: game = safe (9.0)
|   |   |   |   n-wk = 1:
|   |   |   |   |   k-r = 1: game = safe (2.0)      (black king to rook = 1)
|   |   |   |   |   k-r = 2: game = lost (3.0)
|   |   |   |   |   k-r = 3: game = lost (3.0)
|   |   |   k-n = 2:
|   |   |   |   n-wk = 2: game = safe (9.0)
|   |   |   |   n-wk = 3: game = safe (9.0)
|   |   |   |   n-wk = 1:
|   |   |   |   |   k-r = 2: game = lost (3.0)
|   |   |   |   |   k-r = 3: game = lost (3.0)
|   |   |   |   |   k-r = 1:
|   |   |   |   |   |   r-wk = 1: game = lost (1.0)
|   |   |   |   |   |   r-wk = 2: game = safe (1.0)
|   |   |   |   |   |   r-wk = 3: game = safe (1.0)
|   |   |   k-n = 3:
|   |   |   |   k-r = 2: game = lost (9.0)
|   |   |   |   k-r = 3: game = lost (9.0)
|   |   |   |   k-r = 1:
|   |   |   |   |   r-wk = 1: game = lost (3.0)
|   |   |   |   |   r-wk = 2: game = safe (3.0)
|   |   |   |   |   r-wk = 3: game = safe (3.0)
```

Figure 10 ID3 chess decision tree

```
                                                r>>n = f -> game = safe
                                                r>>k = f -> game = safe
                                                line = f -> game = safe
                                     k-n = 1 & n-wk = 2 -> game = safe
                                     k-n = 1 & n-wk = 3 -> game = safe
                                     k-r = 1 & r-wk = 2 -> game = safe
                                     k-r = 1 & r-wk = 3 -> game = safe
                                     k-n = 2 & n-wk = 2 -> game = safe
                                     k-n = 2 & n-wk = 3 -> game = safe
 k-r = 2 & n-wk = 1 & line = t & r>>k = t & r>>n = t -> game = lost
 k-r = 3 & n-wk = 1 & line = t & r>>k = t & r>>n = t -> game = lost
n-wk = 1 & r-wk = 1 & line = t & r>>k = t & r>>n = t -> game = lost
 k-n = 3 & r-wk = 1 & line = t & r>>k = t & r>>n = t -> game = lost
  k-n = 3 & k-r = 2 & line = t & r>>k = t & r>>n = t -> game = lost
  k-n = 3 & k-r = 3 & line = t & r>>k = t & r>>n = t -> game = lost
```

Figure 11 C4.5 chess rules

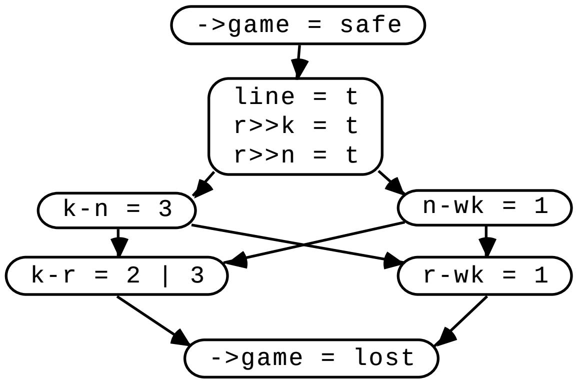


Figure 12 C4.5 chess rules with exception, factored as EDAG

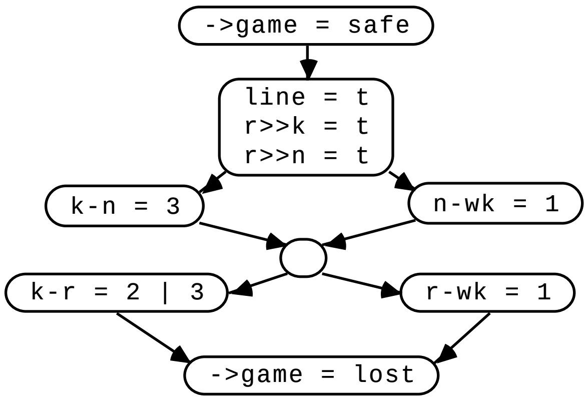


Figure 13 C4.5 chess rules with exception, factored with crossing reduction

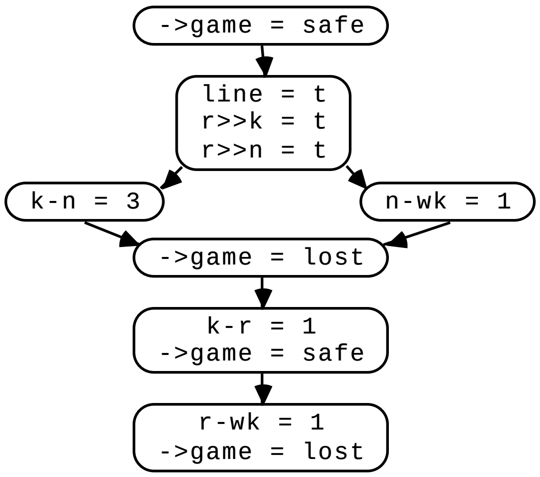


Figure 14 Induct chess rules with exceptions, factored as EDAG

Fig.	Example	Nodes N	Final Nodes F	Arcs A	Excess E = A +F-N	Clauses C	Complexity (N+2E+2C) /5
-	Cendrowska tree	78	52	76	50	130	87.6
-	PRISM production rules	15	15	0	0	73	32.2
10	ID3 decision tree	30	20	28	18	50	33.2
11	C4.5 production rules	15	15	0	0	70	31.0
-	PRISM default, factored	9	1	11	3	12	7.8
12	C4.5 default, factored	7	1	9	3	10	6.6
13	C4.5 default+, factored	8	1	9	2	10	6.4
14	Induct exceptions, factored	7	1	7	1	11	6.2

Figure 15 Comparison of complexity of chess EDAGs

7 Inference with EDAGS—Unknown Values

Inference with EDAGs when all premises are decidable is simple and has already been described. However, inference when some premises are unknown as to their truth values involves some subtleties that are significant. Consider the EDAG of Figure 16 where Premise-1 is unknown but Premise-2 is true and the attached conclusion is the same as the default. One can then infer Conclusion = A regardless of what the truth value of Premise-1 might be.

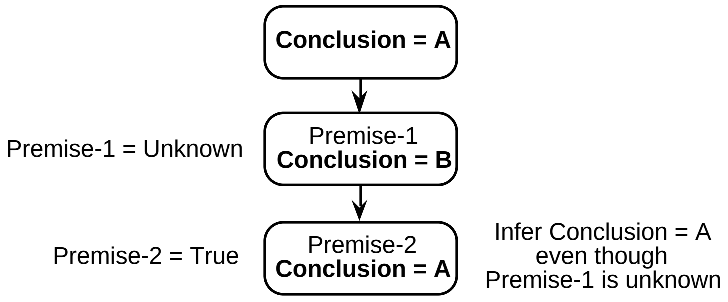


Figure 16 Inference with unknown values

The required reasoning is simple to implement and is incorporated in the EDAG-inference procedure of KRS (Gaines, 1991a; Gaines, 1993a), a KL-ONE-like knowledge representation server. The evaluation of a premise as true, false or unknown is common in such systems. It is simple to mark the EDAG such that a node is disregarded if: it has an unknown premise but has a child node that is definitely true; or it has the same conclusion as another node whose premise is true.

The importance of pruning the list of possible conclusions is that it is the basis of acquiring further information, for example, by asking the user of the system. It is important not to request information that may be expensive to obtain but is irrelevant to the conclusion. Cendrowska (1987) makes this point strongly in comparing modular rules with decision trees; that, for example, in contact lens prescription the measurement of tear production is time-consuming and unpleasant, and should be avoided if possible. She notes that it is not required in the 3 rules produced by PRISM shown at the middle left of Figure 3 but is the first attribute tested in the ID3 decision tree of Figure 2. She argues that the tree requires the testing of this attribute unnecessarily in certain cases.

However, the three decidable cases with unknown values of tear production (corresponding to the premises of the three rules at the middle left of Figure 3) are all correctly evaluated by the EDAG of Figure 2 using the reasoning defined above, as they are by all the other EDAGs of Figures 4 through 7. The problem Cendrowska raises is a question of properly using the tree as a basis for inference, rather than a distinction between trees and production rules.

Since the complexity figures in Figure 9 indicate that the PRISM rules are, on one reasonable measure, more complex than the ID3 tree, it would seem that the arguments for rules being better than trees are not justified. Certainly the highly restrictive standard decision tree can be improved in comprehensibility by the use of exceptions and factoring, but whether the resultant structure is a more general rule, or a set of rules with exceptions, is a matter of perspective.

The simple pruning procedure described above is sufficient to deal with the tear production problem raised by Cendrowska. However, it is inadequate to properly account for all the nine decidable cases with unknown values (corresponding to the premises of the rules in Figure 3). For example, someone whose tear production is normal, astigmatism is astigmatic and age is young but whose prescription is unknown should be inferred as lens is hard. However, this inference cannot be made with the ID3 tree of Figure 2 using the procedure described above alone. KRS copes with this situation by keeping track of the relation between child nodes that are such that one of them must be true. In the case just defined it infers that the conclusion is lens is hard because the two nodes with lens = hard conclusions at the lower right of Figure 2 are both possible, one of them must be true, and both have the same conclusion.

KRS also disregards a node if the conclusion is already true of the entity being evaluated, again to prevent an unnecessary attempt to acquire further information. The four strategies described are such as to reduce the list of possible conclusions to the logical minimum, and form a complete inference strategy for EDAGs.

It should be noted that the completeness of inference is dependent on the possibility of keeping track of relations between child nodes. This is simple for the classification of single individuals based on attribute-value data. It is far more complex for EDAG-based inference with arbitrary KL-ONE knowledge structures involving related individuals, where it is difficult to keep track of all the relations between partially open inferences. This corresponds to managing the set of possible extensions of the knowledge base generated by resolving the unknown premises, and is inherently intractable.

8 EDAG Knowledge Discovery Tools

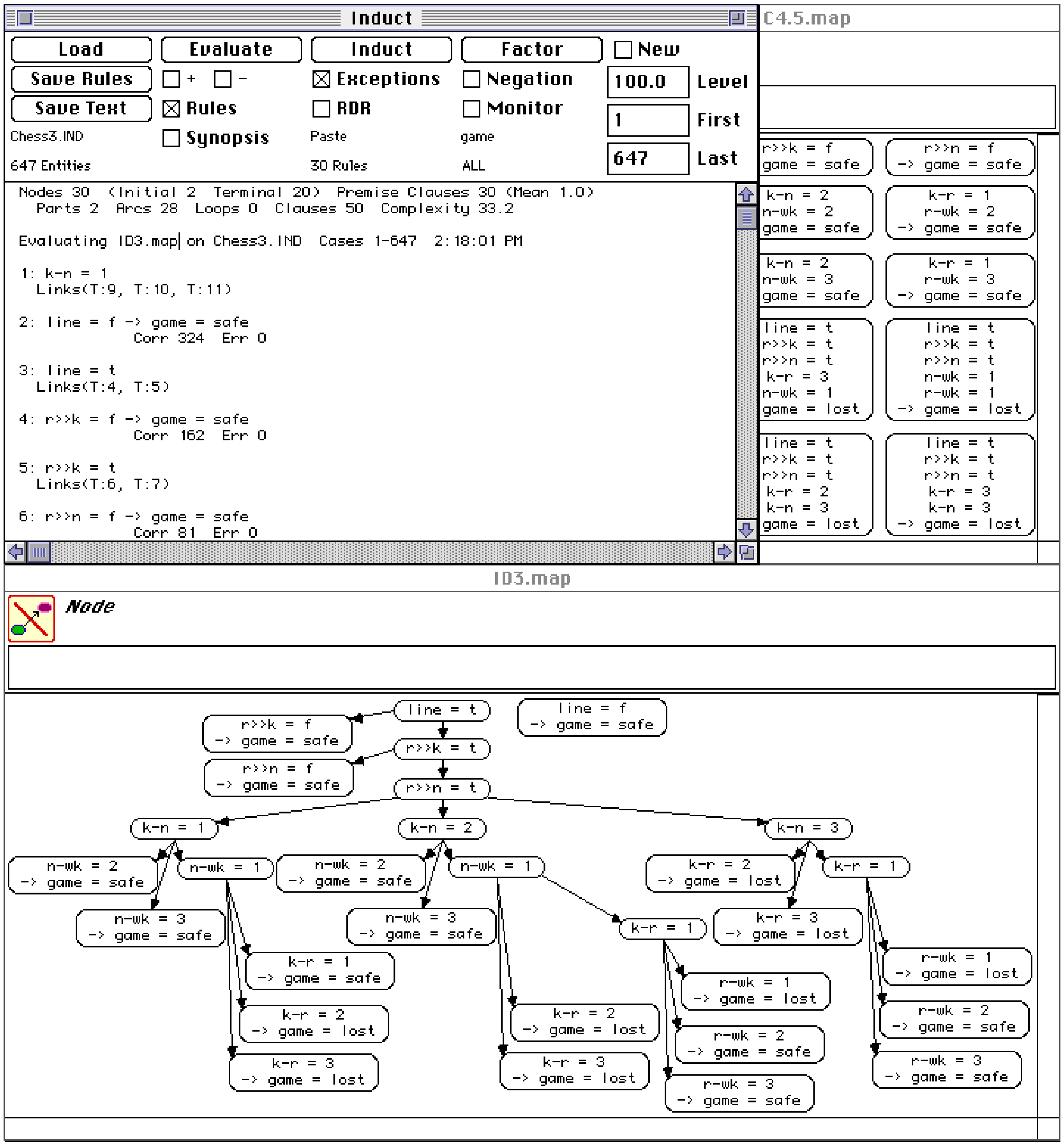
EDAGs are simple to support through visualization and graphic editing tools interfaced to induction and evaluation systems. Figure 17 shows the Induct and KMap tools in KSSn (Gaines, 1994) being used to edit and evaluate EDAGs for the chess end game discussed above.

At the top left is an Induct server window with the chess data loaded. At the top right is a partial view of a KMap window with the C4.5 rules of Figure 11 represented graphically, and at the bottom is a KMap window with the ID3 rules of Figure 10 represented graphically.

KMap is a general visual language tool for knowledge structures (Gaines, 1991c) and concept maps (Gaines and Shaw, 1994). Induct exports its derived EDAGs as graphs in KMap, and edited graphs in KMap may be pasted back into Induct for evaluation of error rates.

The text in the bottom part of the Induct window at the top left of Figure 17 was generated by copying the EDAG in the KMap window at the bottom, pasting it into the Induct window, and then clicking the “Evaluate” button to cause Induct to evaluate the performance of the EDAG on the chess data.

The graphic interface between KMap and Induct makes it simple for an expert or knowledge engineer to experiment with the induced knowledge structures, investigate the roles of different parts, and attempt to improve the comprehensibility of what has been derived.



☐ New
☐ + ☐ -
☒ Exceptions
☐ Negation
☒ Rules
☐ RDR
☐ Monitor
☐ Synopsis

Chess3.IND	Paste	game	100.0 Level
647 Entities	30 Rules	ALL	1 First
			647 Last

```
Nodes 30  (Initial 2  Terminal 20)  Premise Clauses 30 (Mean 1.0)
  Parts 2  Arcs 28  Loops 0  Clauses 50  Complexity 33.2

Evaluating ID3.map on Chess3.IND  Cases 1-647  2:18:01 PM

1: k-n = 1
  Links(T:9, T:10, T:11)

2: line = f -> game = safe
             Corr 324  Err 0

3: line = t
  Links(T:4, T:5)

4: r>>k = f -> game = safe
             Corr 162  Err 0

5: r>>k = t
  Links(T:6, T:7)

6: r>>n = f -> game = safe
             Corr 81  Err 0
```

Figure 17 Graphic EDAG induction and editing tools

9 Discussion and Conclusions

The factorization of systems of rules with exceptions is proposed as a means for simplifying induced rules to derive more comprehensible knowledge structures. The resultant structure is an example of a knowledge structure, an exception directed acyclic graph, that subsumes both trees and rules. EDAGs make apparent the conceptual structures underlying inference through inheritance and structure sharing. EDAG inference is computationally simple, even under conditions of partial information, and its simplicity should make it easily understood by people.

The EDAGs studied in this article are simple and leave open the question of the scalability of the methodology to more significant problems. A recent study (Gaines, 1995) has used EDAGs to reconstruct Shapiro's (1987) "structured induction" of a solution to the rook version pawn end-game when the pawn is about to queen. Replacing the ID3 trees with EDAGs results in a very much smaller solution to the problem that is more comprehensible. In addition, Induct directly induces an EDAG for the complete problem with a structure similar in size and conceptual relations to the structured induction solution.

The EDAGs studied in this article have all related to noise-free data. However, C4.5 and Induct are both effective with noisy data, and can be used to generate EDAGs with probabilistic conclusions.

A number of studies have developed graph-like decision structures that relate to EDAGs. Compton and Jansen's (1990) ripple-down rules generalize binary decision trees by allowing a node to contain a compound premise, and interior nodes to contain conclusions which are asserted if the tree cannot be traversed further. Gaines (1991) shows that ripple-down rules are a particular case of rules with exceptions that can encode some knowledge structures more compactly. Oliver (1993) and Kohavi (1994) have shown how various forms of decision graphs may be induced and provide a more compact alternative than decision trees.

Many questions remain open. Psychological studies of the nature of comprehensibility of knowledge structures are necessary to give substance to the intuitions that lie behind the developments reported in this paper. The current EDAG structure may be too 'flat' in that there is no means to lift the same knowledge component out of different parts of the graph.

All that can be said currently is that the approach is promising and provides a synthesis of trees and rules that goes beyond both and may indicate an interesting new avenue of research and development in knowledge discovery systems.

Acknowledgements

This work was funded in part by the Natural Sciences and Engineering Research Council of Canada. I am grateful to Paul Compton and Ross Quinlan for access to their research and for discussions that have influenced this work. I am also grateful to the anonymous referees for helpful comments. I am grateful to Padhraic Smyth for a detailed critique of the original paper which has greatly improved it.

References

Cendrowska, J. 1987. An algorithm for inducing modular rules. *International Journal of Man-Machine Studies* 27(4) 349-370.

Clancey, W.J. 1987. From GUIDON to NEOMYCIN and HERACLES in twenty short lessons. *Current Issues in Expert Systems*. 79-123. London: Academic Press.

Clancey, W.J. 1989. Viewing knowledge bases as qualitative models. *IEEE Expert* 4(2) 9-23.

Clancey, W.J. 1993. The knowledge level reinterpreted: modeling socio-technical systems. *International Journal of Intelligent Systems* 8(1) 33-49.

Compton, P. and Jansen, R. 1990. A philosophical basis for knowledge acquisition. *Knowledge Acquisition* 2(3) 241-258.

Gaines, B.R. 1977. System identification, approximation and complexity. *International Journal of General Systems* 2(3) 241-258.

Gaines, B.R. 1989. An ounce of knowledge is worth a ton of data: quantitative studies of the trade-off between expertise and data based on statistically well-founded empirical induction. *Proceedings of the Sixth International Workshop on Machine Learning*. 156-159. San Mateo, California: Morgan Kaufmann.

Gaines, B.R. 1991a. Empirical investigations of knowledge representation servers: Design issues and applications experience with KRS. *ACM SIGART Bulletin* 2(3) 45-56.

Gaines, B.R. 1991b. Integrating rules in term subsumption knowledge representation servers. *AAAI'91: Proceedings of the Ninth National Conference on Artificial Intelligence*. 458-463. Menlo Park, California: AAAI Press/MIT Press.

Gaines, B.R. 1991c. An interactive visual language for term subsumption visual languages. *IJCAI'91: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*. 817-823. San Mateo, California: Morgan Kaufmann.

Gaines, B.R. 1991d. Refining induction into knowledge. *Proceedings of the AAAI Workshop on Knowledge Discovery in Databases*. 1-10. Menlo Park, California: AAAI.

Gaines, B.R. 1993a. A class library implementation of a principled open architecture knowledge representation server with plug-in data types. *IJCAI'93: Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*. 504-509. San Mateo, California: Morgan Kaufmann.

Gaines, B.R. 1993b. Modeling practical reasoning. *International Journal of Intelligent Systems* 8(1) 51-70.

Gaines, B.R. 1994. Class library implementation of an open architecture knowledge support system. *International Journal Human-Computer Studies* 41(1-2) 59-107.

Gaines, B.R. 1995. Inducing knowledge. *Proceedings of the Ninth Knowledge Acquisition for Knowledge-Based Systems Workshop*. to appear. Alberta: University of Calgary.

Gaines, B.R. and Shaw, M.L.G. 1994. Concept maps indexing multimedia knowledge bases. *AAAI-94 Workshop: Indexing and Reuse in Multimedia Systems*. 36-45. Menlo Park, California: AAAI.

Kohavi, R. 1994. Bottom-up induction of oblivious read-once decision graphs: strengths and limitations. *AAAI'94: Proceedings of the Twelfth National Conference on Artificial Intelligence*. 613-618. Menlo Park, California: AAAI Press/MIT Press.

Li, X. 1991. What's so bad about rule-based programming? *IEEE Software* 103-105.

Nosek, J.T. and Roth, I. 1990. A comparison of formal knowledge representations as communication tools: predicate logic vs semantic network. *International Journal of Man-Machine Studies* 33 227-239.

Oliver, J.J. 1993. Decision graphs - an extension of decision trees. *Proceedings of 4th International Workshop on AI and Statistics*. 343-350.

Quinlan, J.R. 1979. Discovering rules by induction from large collections of examples. *Expert Systems in the Micro Electronic Age*. 168-201. Edinburgh: Edinburgh University Press.

Quinlan, J.R. 1991. Foreword. *Knowledge Discovery in Databases*. ix-xii. Cambridge, Massachusetts: MIT Press.

Schreiber, A.T., Wielinga, B.J. and Breuker, J.A., Ed. 1993. *KADS: A Principled Approach to Knowledge-based System Development*. London: Academic Press.

Shapiro, A.D. 1987. *Structured Induction in Expert Systems*. Wokingham, UK: Addison-Wesley.

Shum, S.B. and Hammond, N. 1994. Argumentation-based design rationale: what use at what cost? *International Journal of Human-Computer Studies* 40(4) 603-652.