An Alternative Verification and Validation Technique for An Alternative Knowledge Representation and Acquisition Technique.

Debbie Richards and Paul Compton

Department of Artificial Intelligence
School of Computer Science and Engineering
University of New South Wales
Sydney, Australia
Email: {debbier, compton}@cse.unsw.edu.au

Abstract: Ripple-Down Rules (RDR) are an alternative from mainstream approaches to the building of knowledge based systems (KBS). RDR use simple, but reliable, techniques for Knowledge Acquisition (KA) and representation which have been shown to support the on-line development, maintenance and validation of KBS. Key features of RDR that affect the approach to V&V are the incremental nature of KA and maintenance, the use of cases for KA and validation, the use of an exception structure for knowledge representation and the development of KBS by experts. This paper describes RDR and its approach to V&V particularly concentrating on recent extensions which use Rough Set Theory for verification and Formal Concept Analysis for validation.

Keywords: Verification and Validation, Ripple-Down Rules, Rough Set Theory, Formal Concept Analysis

1 A Different Approach to KA

Many approaches to knowledge based system (KBS) development attempt to build complete systems that are mostly considered final before being put into routine use. These approaches are based on Newell's (1982) Knowledge Level which advocates the modeling of knowledge at a level above its symbolic representation and includes modeling of problem solving methods (Chandrasekaran 1986, McDermott 1988, Puerta et al 1992, Schreiber, Weilinga and Breuker 1993 and Steels 1993) and ontologies (Guha and Lenat 1990, Patil et al 1992 and Pirlein and Studer 1994). The need for complex modeling as a prerequisite to knowledge acquisition (KA) has resulted in the development of verification (Cragun and Streuduel 1987, Preece, Shinghal and Batarekh 1992, Suwa, Scott and Shortliffe 1982) and validation (O'Keefe and Leary 1993) (V&V) techniques that are designed for use before the system goes into routine use. There is little consideration of incremental validation of such systems and maintenance is often a neglected problem (Kang, Gambetta and Compton 1996, Menzies and Compton 1995 and Soloway, Bachant and Jensen 1987). Santos, Gleason and Banks (1997) have found that "the techniques of knowledge-base validation have been left open for interpretation" (Santos, Gleason and Banks 1997, p.14) and that they lack "historical benchmarks". Validation of systems is often left to vague heuristics such as: If the result is similar to a human expert then it is okay. Perhaps validation poses such a problem due to the difficulty of trying to validate models that are by their very nature inaccurate (Clancey 1991). Another reason for trying to get the system *right* at the start is that in standard production rule systems changes do not occur locally and even small changes can result in major effects that are not easily resolved (Gaines 1991, Li 1991).

To avoid the problem of side-effects that occur when maintaining a typical production rule KBS, rules are never changed or deleted in an RDR KBS (Compton and Jansen 1990). When the expert determines that a case is misclassified a new rule is added as a refinement to the rule that gave the wrong conclusion. This new rule is reached only if the same sequence of rules is followed. The utility of RDR has been demonstrated by Pathology Expert Interpretive Reporting System (PEIRS) (Edwards et al 1993) which went into routine use in a large Sydney hospital with about 200 rules and over a four year period (1990-1994) grew online to over 2000 rules. Approximately 100 hours was spent on adding rules to the PEIRS system. Current evaluations of a commercial system have shown experts can build systems with 3- 4,000 rules in about one person week.

RDR does not distinguish between initial KA and system maintenance. RDR develops the whole system on a case by case basis and automatically structures the KBS in such a way to ensure changes are incremental. Validation is designed to be performed on-line by the expert. Other KA approaches such as repertory grids (Gaines and Shaw 1989) also minimise the role of the knowledge engineer but like most KA approaches seek to define globally applicable rules and to develop a stereotypical set of cases which require some consideration of the whole domain and do not consider incremental maintenance. With repertory grids, maintenance of the rules is addressed by regenerating implications associated with the revised data set.

The difference between the simple user-driven RDR approach to KA and mainstream approaches that require complex modelling performed by a knowledge engineer hinges on the fundamental differences in the philosophical foundations of these approaches. RDR does KA with minimal analysis (Compton et al 1993) on the basis that experts use what is in their head to "make up" a solution to fit the situation. The capture of behavioural knowledge based on cases appeared to offer a solution which avoided trying to understand what went on inside the experts head.

2. A Different Approach to Knowledge Representation

In single classification RDR we define an RDR as a triple <rule,X,N>, where X are the exception rules and N are the if-not rules (Scheffer 1996). When a rule is satisfied the exception rules are evaluated and none of the lower rules are tested. An extension of RDR has been made to cover domains that require multiple independent classifications. Multiple Classification RDR (MCRDR) has been shown to build KBS that mature more quickly and are more compact than single classification RDR KBS even for single classification domains (Kang, Compton and Preston 1995, Kang 1996). It is conjectured that this occurs because more use is made of expertise rather than depending on the KB structure (Kang 1996). An MCRDR is defined as the quadruple <rule,P, C,S>, where P is the parent rule, C are the children rules and S are the siblings. All siblings at the first level are evaluated and if true the list of children are evaluated until all children from true parents have been exhausted. The last true rule on each pathway form the conclusions for the case. Figure 1 shows an MCRDR KBS.

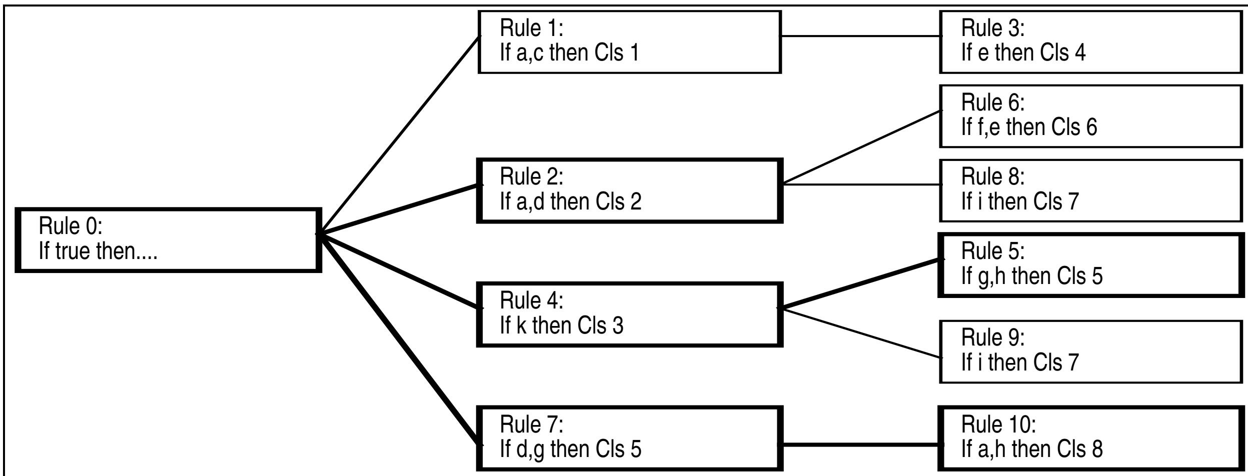


Figure 1. An MCRDR KBS. The highlighted boxes represent rules that are satisfied for the case {a,d,g,h,k}. We can see that there are three independent conclusions for this case, Class 2 (Rule 2), Class 3 (Rule 4) and Class 8 (Rule 10).

RDR are a knowledge representation (KR) and acquisition technique that non-monotonically reasons over a list of ordered rules with exceptions. RDR belong to a class of representation schemes that provide modularity through the use of exceptions. This class also includes Dimopoulos and Kakas 1995, Grosof 1997, Helmbold, Sloan and Warmuth 1989, Kivinen, Mannila and Ukkonen 1993, Kowalski and Sadri 1991, Rivest 1987 and Vere 1980. Of these representations, RDR are "the most general for attribute-value based representations and most strongly structured" (Scheffer 1996, p. 279). This interest in the use of exceptions is based on a number of useful features that exceptions support. The key benefits offered by exceptions are: the naturalness and comprehensibility of the representation by human experts; the support they offer for incremental change; the use of context and the maintainability of the knowledge due to the localisation of changes (Gaines 1991, Li 1991). Exceptions also provide a compact representation and accounts for the use of exceptions as output structures by some machine learners (Vere 1980). The size of the KBS can be further reduced through the use of a default head rule with no conditions which fires if all other rules fail. This avoids the need to specify the class for impossible combinations of attributes (Catlett 1992).

The value of local refinements is evident in the work by Grosof (1997) on Courteous Logic Programs (LP). Courteous LP use prioritised defaults, which respect precedence, and even when conflict exists they provide a consistent, tractably computable and unique set of conclusions. In addition Courteous Logic avoids many of the pitfalls of the use of negation-as-failure by using classical negation, defaults and priorities (which are pairwise override tags associated with rules). Grosof points out that "a key to courteous LP's computational and conceptual simplicity is that the prioritization and conflict resolution is local, pairwise and atomic" (Grosof 1997, p.13).

Local refinement aids incremental development and the use of exceptions tends to avoid some of the verification anomalies from which non-ordered rule bases suffer. Many exception KRs are nonrecursive and are made up of rules which go straight from condition to conclusion unlike most production rule system which use intermediate rules. These features avoid the verification anomalies identified by Nguyen et al (1987) of circular rules, unachievable intermediate conclusions, unachievable (final) conclusions or premises.

3 A Different Way of Using Cases

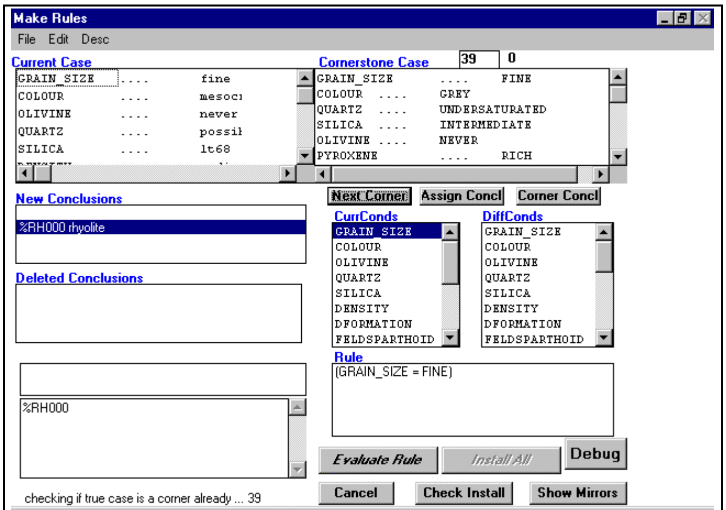


Figure 2: The MCRDR screen for making rules

RDR handles the issue of context by its exception structure and the storing of the case that prompted a rule to be added. This case is known as the cornerstone case and assists the expert in KA and validation of new rules. In Figure 2 we see the MCRDR for Windows screen for making new rules. First, the user selects the conclusion. They then specify the attribute-value pairs that form the rule conditions. A difference list is shown to the user as conditions must differentiate the current case from the cornerstone cases associated with the rule that gave the misclassification (Compton & Jansen 1990). In this way the rule is validated online against the case when it is added. In single classification RDR only one case is associated with each rule. In MCRDR there may be multiple cases that must be distinguished from the current case. In the KA approach developed, the expert is presented with one cornerstone case at a time. The expert constructs a rule to distinguish the new case from the first case presented and then each case in the cornerstone list is evaluated to see if it is also distinguished by the new rule. If a case is satisfied by the rule the expert must add extra conditions to the rule to distinguish this case. This continues until all related cornerstone cases are distinguished. Remarkably the expert provides a sufficiently precise rule after two or three cases have been seen (Kang, Compton and Preston 1995). The user may decide to stop an incorrect conclusion instead of replacing it with a new conclusion. This is achieved in the same way by adding a stopping rule which has a null conclusion.

The rule which the expert develops becomes the index by which the case is retrieved. The KA technique provides a mechanism for storing, retrieving and differentiating cases. The RDR approach provides a solution to a common problem associated with typical case based reasoners which is how to index cases for retrieval and how to adapt them to fit the new case (Kolodner 1993).

4. A different approach to V&V

As implied in the previous sections, RDR are more than a knowledge representation. RDR are a hybrid rule-based and Case Based Reasoning (CBR) technique where cases are used to motivate and validate KA and rules are used to index the cases for storage and retrieval. It is thus the combination of the exception structure and the use of cases which makes RDR a somewhat different approach to many other approaches which may also make use of exceptions for knowledge representation. In this paper we focus on the V&V of RDR KBS but many of the techniques are applicable to other rule representations that use the form variation 1, variation 2, …variation n $\rightarrow$ Action.

To verify our KBS we must ensure that the constraints of the KR and the inferencing mechanism are not violated (Santos, Gleason and Banks 1997). The exception structure and the use of cases provide constraints which ensure that the pathway for each case is maintained and that each case is differentiated from previous cases which have similar features but warrant different conclusions. The nonmonotonic nature of our KBS, where each prior conclusion may be replaced by another conclusion when new evidence is presented, requires a modified V&V approach to conventional monotonic rule bases. Other features of RDR which affect the approach to V&V are the incremental nature of KA and maintenance, the use of cases for KA and validation, the use of an exception structure for knowledge representation and the development of KBS by experts.

Verification has traditionally been broken up into basically two categories: consistency and completeness (Gonzalez and Dankel 1993). With Santos, Gleason and Banks (1997) we have found that the "push for *consistency and completeness* as the goal of verification is an artifact of rule-based inferencing mechanisms, and not necessarily a requirement for all knowledge-base verifications" (Santos, Gleason and Banks 1997, p.14). Santos, Gleason and Banks are interested in providing an incremental V&V technique and argue that if we tolerate or even expect our KBS to be incomplete we can not enforce the constraint of completeness. While the conventional list of verification anomalies can not be ignored the full list does not apply to the RDR exception structure. Also the approach to the detection of such anomalies and the decision of what should be done about them may differ. For example, repetition in an RDR KBS is unavoidable as knowledge is patched locally and the same patch may need to be applied in multiple locations.

Not all approaches treat V&V as a phase that occurs before a system is deployed. Chander, Shinghal and Radhakrishnan (1997) point out that KBS tend to evolve rather than be developed as a complete system. This means that incremental evaluation is important. One of the problems is that many techniques, such as regression testing, look at evaluating the whole KBS each time a change is made. This problem also occurs when validation is performed by generating partially ordered rule sequences called paths (Grossner et al 1996) where "adding one more rule could result in changed paths, thus altering the goal relevancy status of the system" (Chander, Shinghal and Radhakrishnan 1997, p.43). This is a time consuming and resource expensive option (Meseguer 1992). Chander, Shinghal and Radhakrishnan (1997) propose the extraction and incremental validation of only those paths that are affected by the change where it is assumed that the goal has not changed. Similar to regression testing is the use of test cases for validation. Barr (1997) has shown the value and importance of having cases that exercise each conclusion, node and class and advocates building a directed graph of rule pathways that tests this has occurred. If it has not, then she advocates adding cases to the test set to cover those parts not being exercised.

Boswell, Craw and Rowe (1997) use the KRUST tool to detect unwanted side-effects by generating rule traces. Both KRUST and CLIPS-R (Murphy and Pazzani 1994) work "backward from observed faulty behaviour" (Boswell, Craw and Rowe 1997, p. 63) and require strategies for handling conflict resolution and rule ordering. The rule traces generated form a *knowledge skeleton* which provide additional information regarding possible effects of changes to the rule-base. The steps used by KRUST are blame allocation, refinement generation and refinement selection.

In contrast, validation of RDR pathways is automatic as the exception structure and KA technique does not allow the user to enter a rule that invalidates previous rules or that does not classify and distinguish the current case. RDR uses a failure-driven approach where the expert or machine-learner detects the assignment of an incorrect conclusion which prompts a new rule to be added. We make no apologies that we require an expert to identify incorrect conclusions. When we are dealing with an incremental technique to KBS development and V&V we can not avoid the need for human involvement. Santos, Gleason and Banks (1997) provide the line of reasoning. They state that to automatically identify and fix incomplete knowledge we need to have complete knowledge. The only way out of this catch-22 is to get the human in the link or to use cases that are already classified (which were possibly classified by a human anyway). It is becoming increasingly acknowledged that automated approaches to KA such as ML can be enhanced by combining expertise from human experts (Webb and Wells 1995 and Shiraz and Sammut 1998). We are of the opinion that this is not a weakness in the approach but a reality of the complexity and situatedness of knowledge.

To summarise using the four part framework of: what to validate; when to validate; which methodology; and what criteria (O'Keefe, Balci and Smith 1987):

- we validate that previous cases remain correctly classified and that the new case is differentiated from the previous cases,
- we validate at KA time where the user performs an inference, determines its accuracy, if incorrect they assign the correct conclusion and select the relevant features in the case and rerun the case to verify the right result is achieved,
- the methodology includes the use of an exception structure and stored cases to maintain and provide the context in which the knowledge applies and a KA technique that assists and constrains the user to focus on differences between contexts.

The criteria against which the system is measured is the number of correctly classified cases over the total number of cases. System accuracy should increase as the KBS grows and more cases are seen.

5. Extending Ripple Down Rules and Verification

In this section we consider which verification anomalies are applicable to RDR and describe an extension to the RDR approach to verification which can detect and remove these anomalies. In brief, redundancy exists and is a potential problem. The problem of circularity does not exist and deficiency is regarded as unavoidable. The type of verification that single classification RDR provide is a guarantee of consistency in that a given set of attribute-value pairs will result in one conclusion being recommended. With the advent of MCRDR we have the situation where a given set of attribute-value pairs may result in multiple conclusions. This means we can no longer take consistency for granted and need to consider checks for consistency between classifications. The approach which is described allows the RDR KBS to be developed using the V&V strategies offered by the exception structure and use of cases and at some point in its life to be reorganised to remove verification anomalies. A summary is provided in Table 1. Recent work regarding the removal of repetition, checking for ambivalence (inconsistency) and the detection of deficiencies in the knowledge using Rough Set Theory (RST) (Pawlak 1991) are now described. First we provide an introduction to rough set theory.

Anomaly	Relevant Yes/No	Solution
redundant rules	yes	Compute reducts using RST to remove.
conflicting rules	yes for MCRDR	Compute dependency coefficient using RST to detect. Resolve manually.
subsumed rules	yes	Use FCA to detect. Use RST to remove.
unnecessary premise conditions	yes	Use RST to remove.
circular rule chains	no	
unreferenced attribute values	yes	Use Exceptions and Credentials to detect missing knowledge.
illegal attribute values	no	
unachievable intermedte conclusions	no	
unachievable (final) conclusions	yes	Remedied as cases with new conclusions are seen.
unachievable goals	no	
unachievable premises	no	

Table 1: Nguyen’s Verification Anomalies and their relevance to RDR.

5.1 An Overview of Rough Set Theory

Pawlak (1982) first introduced rough set concepts as a means of discovering relationships in data. RST has been used successfully to support the classification of imperfect data in a range of domains (Fibak, Slowinski and Slowinski 1986, Krysinski 1990 and Slowinski, Slowinski and Stefanowski 1988). Pawlak defines a knowledge base to be a family of classifications[1] in the universe of objects being investigated. Each family member forms a subset or category. Membership of a subset is sometimes uncertain and the concepts of upper and lower approximations, $\overline{R}$X and $\underline{R}$X respectively, are used to approximate membership, where X is the concept and R is the knowledge held. If a concept is known, that is all objects with particular attribute value (A-V) pairs belong to the same classification, then those objects belong to the lower approximation and form the positive region. More formally we define $POS_R(X)$ which says that employing knowledge R the set POS can be certainly classified as elements of X. Those objects

[1]The discussion that follows is restricted to problems of classification.

that certainly don't belong form the negative region. If the information we have is not sufficient to classify an object then such objects belong to the boundary region. The upper approximation is the set theoretical union of the positive and boundary regions. Using these regions, classified sets are used to help approximate the classification of the unclassified set. This replaces the notion that set membership is a primitive relation with the view that the membership relation is based on what knowledge is held.

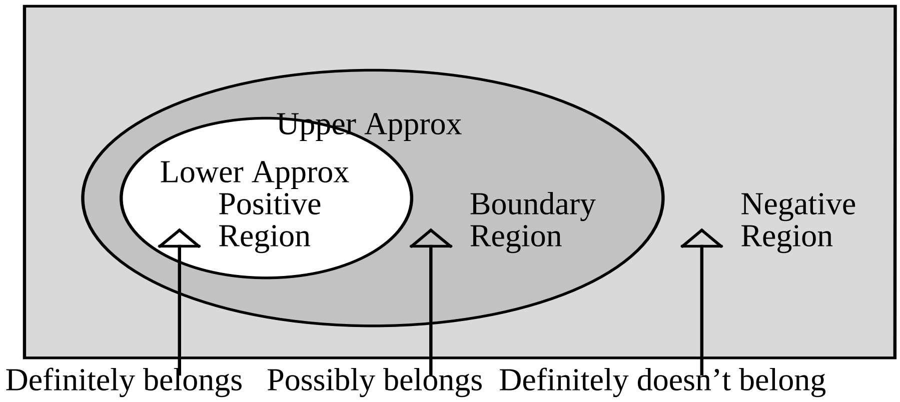


Figure 3: The Partitions in a RST Universe

The regions shown in Figure 3 are used to find dependencies (rules) through the generation of the core and the reducts of a knowledge base. Reducts are collections of attribute value pairs by which a conclusion can be reached and can be seen as alternate rule pathways. The core is the intersection of the reducts. Calculation of the core and reducts can be used to simplify decision tables. This is achieved through the removal of redundant attributes (columns), rules (rows) and superfluous attribute values (Pawlak 1991, p.71). For a comprehensive description of RST and how core and value reducts are used to remove redundancies refer to Pawlak (1991). Previous work done with rough sets has been concerned with analysis of cases rather than rules[2]. A key feature of treating rules as cases is that there are a larger number of missing values in rules than in cases. However, RST appeared suitable for verification of rule sets since it is concerned with the reduction of consistent algorithms through the removal of multiple instances (objects in rough set terminology) and redundant attributes and since is has been particularly designed to handle uncertainty and missing values. Also, RST does not suffer from the same restriction that we found in other ML techniques, such as C4.5 or Induct, where the number of cases affects the rules produced.

5.1.1 PREPARATION OF THE RULES

A necessary prerequisite for applying RST was the conversion of the rule base into decision table format. For rulebases that use intermediate conclusions this process is achievable by exploding each rule into a lower level form where intermediate conclusions are replaced with the primitive conditions. Colomb (1993) provides an algorithm for converting between decision tables, decision trees and propositional KBS. In an MCRDR KBS the conversion process is much simpler and does not result in the "inflation" problem described by Colomb and Sienkiewicz (1995) because rule conditions are always made up of conjunctions. Disjunction is not allowed. Each rule represents a rule pathway which may be treated as a row in a decision table. This feature has also been beneficial in the conversion of MCRDR KBS into formal contexts as described later in section 6.2. Once the rules are in the rough set format (decision table) the iterative process described below and summarized in Figure 4 is performed. We will describe the process using a small (42 rules, 11 attributes and 31 conclusions) MCRDR KBS developed for the SISYPHUS III (Shadbolt 1996)

---

[2]An exception is research by Colomb and Sienkiewicz (1995) which used RST to reduce the "inflation" that can occur when a propositional ES is converted into a decision table.

geology domain. Thus we begin with a decision table with 42 rows and 12 columns, the twelfth column contains the conclusion which is known as the decision attribute in rough set terminology.

```
Convert MCRDR KBS into rough set format (SYS)
While reducts (rules) of SYS show redundancies exist
   Determine why attribute, object or value was removed
   Fix objects
End-While
While dependency coefficent κ < 1 for each individual class and groups of mutually exclusive conclusions
  List all objects not belonging to the positive region and the object that they conflict with.
  Fix objects
End-While
While classification coefficient γ < 1
  Find objects not in the lower approximation or it's complement
  Fix objects
End-While
Output new rules (These may be converted back into the original format or used as tables).

Fix objects.
If coding error, fix.
Else If error due to missing values
        Initialise A-V check table
        Get set of objects that match with the object being investigated
        While there are more objects in the set
          read an object
          update A-V check table with a found flag for each non-missing value
       End-While
       Build one or more new objects from the original object using the AV check table to cover those values not
         covered by other objects.
Update SYS with new objects, modified objects and remove objects in error.
```

Figure 4: An algorithm of the verification process using rough sets. There are three main phases each contained within a while loop. Procedure Fix Objects is expanded to provide more detail. The word rule refers to the original and output rule sets. The word object refers to the cases being used by the RSL which represent the rules.

5.1.2 PHASE ONE: REMOVAL OF REDUNDANCIES

One of the disadvantages of locally patching the RDR exception structure is the existence of repeated and redundant knowledge. The repetition problem, however, has proven to be far smaller than first anticipated. Studies by Mansuri et al (1991) and Compton, Preston and Kang (1995) using real and simulated experts, respectively, have shown that the KBS produced using manual RDR are comparable in size to those built using the machine learning algorithms Induct (Gaines 1989) and C4.5 (Quinlan 1992). Our newer representation, MCRDR, has been shown to exhibit even less repetition (Kang 1996). Despite repetition not being excessive we are interested in removing or at least reducing repeated knowledge for efficiency and particularly for explanation purposes. Gaines (1991) has suggested that, should the repetition problem become large then, the cases associated with that rule base can be used as input to a machine learning algorithm such as Induct/RDR to produce a more compact version of the KBS. Some work (Chellen 1995) has been done to test this out and further work (Richards, Chellen and Compton et al 1996) compared the results using Induct and Rough Set theory (RST) (Pawlak 1991) for rebuilding more compact RDR KBS. The results are given in this section. First we consider the type of redundancy that exists in RDR KBS.

Subsumed rules are an anomaly which can occur in the RDR structure. The use of local patching by exceptions can result in pathways where the rule conditions of one pathway subsume the rule conditions of another pathway and both rules have identical conclusions. This can occur because a rule that, for example, gave the conclusion %VC000-Volcanic has been amended in the context of a new case which contains a feature not in the cornerstone case. We can then get a later situation where a new case should have the original conclusion %VC000-Volcanic because although it also has all the same conditions that form the pathway for the previous exception rule the presence of a different attribute-value pair means that the previous exception is overridden and the amended conclusion does not apply any longer. We also get this situation occurring in different parts of the KBS because the order in which cases are seen will affect the structure and which conditions are chosen first. Excessive subsumption is undesirable but we argue that the history provided by the chain of rules is valuable and also results in the ability to draw the subsumption lattices described in section 6.2.

Another type of redundancy is unnecessary premise conditions which occurs when we have two rules that give the same conclusion but the condition in one negates the condition in the other while all other conditions are the same, as in the example :

Rule 1: If A=X and B=Y then C=Z   Rule 2: If A=X and B=NOT Y then C=Z

The second condition in both rules is unnecessary and both rules can be combined to form the one rule:

If A=X then C=Z.

RST defines redundancy in the same way as Preece, Shinghal and Batarekh (1992). If we remove a column, row or value without affecting the hypotheses derived then it is termed *dispensable* and can be removed. To find redundancies using rough sets we input the original set of rules in decision table format to get a new set of reduced rules (reducts) which have had dispensable objects, attributes or attribute-value pairs removed. We continue our discussion using a KBS developed from the SISYPHUS III data that was converted into a matrix in section 5.1.1.

Through the generation of reducts, redundancy in the 42 objects in our decision table was removed and resulted in 40 reduced rules. These two rules were removed either because they were originally duplicates (with conditions occurring in a different sequence but giving the same conclusion) or they became duplicates after dispensable conditions had been removed. The latter situation was the reason for the rules being dropped from the SISYPHUS III KBS. While only two rules were removed we had a substantial decrease in size of the rule base in terms of the number of conditions which went from a mean number of 4.0 conditions per rule to a mean number of 2.2 conditions per rule. The reduced KB was then evaluated against the cornerstone cases to ensure that each case gave the same set of original conclusions (remember we are dealing with multiple classification RDR). It was found that five of the fifteen cornerstone cases gave one or two extra conclusions. By making the rules more general there were more cases covered by a rule. Assuming that our original classifications were right, it was necessary to make some rules more specific to preclude the misclassified cases by adding the A-V pair (rule condition) that had been removed. If more than one condition had been removed then by looking at the case it could be determined which condition/s needed to be added. The high number of overclassified cases is initially alarming but if we take into account the cases had an average of 6.2 conclusions each and that for a total of 93 original conclusions we obtained 99 conclusions, we achieved 0.94% accuracy. To achieve 100% accuracy on the cornerstone cases four rules out of the 40 needed one extra condition added.

In addition to the small experiment using the SISYPHUS III KBS we have conducted much larger experiments using KBS built by simulated experts[3]. A number of KBS were built based on randomised datasets from the Irvine Machine Learning Repository for the Garvan, Chess and Tic-Tac-Toe domains. We used two different machine learning algorithms, RST and Induct (Gaines 1989) to see whether we could reorganise the KBS into smaller KBS with no loss in classificatory accuracy. We used two different types of cases as input. One input type used all the cases as is and the other used a cleanup algorithm on the cases. Our motivation for using a cleanup algorithm before using the cases for ML is due to the fact that cases often contain attributes that are irrelevant in determining the conclusion. These attributes can misguide the machine learner or at least reduce the learners efficiency in acquiring the optimum set of rules. The clean-up process involved taking a set of cornerstone cases, a set of rules and running each case against the rules keeping for each case only those attributes that caused a rule to either fire or those that caused a rule to misfire, while replacing all other attributes by missing values. This method of combining cornerstone cases and rules and creating a new *case* with relevant A-V pairs will be referred to as Cleanup 1.

An alternative approach, known as Cleanup 2, applied to the handling of conditions on the false branch and attempted to generalise and replace a value that caused a rule to fail with the negation of the value in the rule. This meant that if the possible values were 1-4, the value in the case was 1 and the value in the rule was 2, then the value in the cleaned-up case would be -2 and the values 1,3 or 4 would be treated as matches. This was seen as a better interpretation of the knowledge. It was assumed that when the expert gives the attribute in the rule a particular value, the expectation is that any cases with all values except for the one specified will fail. Where only two values were possible for an attribute and the condition evaluated to false the value in the case was taken. This is the same as Cleanup 1. To reduce complexity and facilitate the techniques described as Cleanup method 1 and 2, the initial KB were converted into a format where attributes in rule conditions only had positive values and negation was not present.

Filtered sets of cases, using Cleanup 1 and Cleanup 2, were then used as the training sets for Vinduct and RS/RDR[4]. This resulted in a set of reduced knowledge bases. The percentage error of the original knowledge base and the compacted knowledge base on the test set was then determined. Plots of the original knowledge base size and compacted knowledge base size against the percentage of total cases used as the training set were obtained for each randomised dataset of each domain. The percentage error of the original knowledge base and the percentage error of the compacted knowledge base versus the percentage of total cases used as the training set were also plotted.

All of the results we achieved on cornerstone cases showed higher error rates when Cleaned Up cornerstone cases, rather than straight cornerstone cases, were used. Cleanup 1 gave better results than Cleanup 2. However neither method was satisfactory and it was necessary to try a different approach. Instead of using the cornerstone cases in the clean up process, the original cases from the dataset were combined with the rules generated by the simulated expert to produce a new set of Cleaned Up cases. For example, rather than compacting the 88 cornerstone cases and 88 rules generated by the simulated expert to cover the first 545 cases (2.5% of the total KBS see Table 2), the 545 cases were compacted using the 88 rules and the 545 Cleaned Up cases were used as input.

---

[3] The use of a simulated expert is described in Compton, Preston and Kang (1995).

[4] Vinduct and RS/RDR are our implementations of the Induct algorithm and rough set theory. They both take RDR trees as input and produce RDR trees as output.

Table 2[5] shows that when the algorithms are applied to the original cases, RS/RDR produces minimally more compact KB than Induct/RDR but produces more errors until 50 percent of cases are seen and then performs better than both Induct/RDR and the Initial KB. At 25% of cases RS/RDR produces less errors than Induct/RDR on cleaned up cases and has less errors than both ML algorithms when used on straight cases. Both algorithms produce larger KBs with Cleaned Up cases than with cases alone, with Vinduct producing smaller rule sets than RST. The error rates for Induct/RDR and Vinduct are about 2% with and without Cleaned Up cases and therefore about 1% higher than the Initial KB. RS/RDR error rates drop using Cleaned Up cases as opposed to straight cases, showing a marked improvement over the Initial KB after 50 percent of cases are seen. The error rates for both algorithms are much closer to those in the Initial KB and suggests that this is a feasible approach to removing repetition from KBS that are incomplete and where a sufficiently large and complete set of cases is not yet available for machine learning. The overall improved results using all cases seen so far rather than just cornerstone cases supports Brian Gaines suggestion (personal communication) that all cases, not just cornerstones, seen by a knowledge base should be kept.

Percent of Cases used to learn	Initial KB #rules	Initial KB %error	Straight Cases #rules		Straight Cases %error		Cleanedup Cases #rules		Cleanedup Cases %error	
			Induct	RST	Induct	RST	Induct	RST	Induct	RST
2.5	88	11.23	34	22	11.69	23.25	45	51	10.13	13.41
5.0	138	9.43	59	35	8.33	20.18	72	83	8.47	10.68
15.0	295	4.89	128	117	5.73	8.96	156	172	6.73	6.12
25.0	402	3.39	171	165	4.34	5.00	203	241	5.22	4.19
50.0	624	1.99	268	226	2.42	4.12	318	343	3.48	1.69
75.0	796	1.26	296	281	1.96	0.86	363	440	2.01	0.56
100.0	958	0.86	358	328	0.00	0.00	428	503	0.05	0.11

Table 1: Compaction on the Garvan Data using Cases rather than Cornerstone Cases.
The first column shows the percentage of the 21822 Garvan cases used in each test. The number of rules and percentage of errors for the initial KB are given in the second and third columns, respectively. All testing was performed on the last 25 percent of the 21822 Garvan cases. The same rules and cases were used by Vinduct and RS/RDR to generate a reduced set of rules. Columns four and five give the number of rules and percentage of error, respectively, produced from using the number of cases indicated in the first column and the straight Induct/RDR or RST algorithms. Columns six and seven show the number of rules and percentage of errors, respectively, produced by Vinduct and RS/RDR when the number of cases in column one are compacted using the rules in column two as input.

The results produced by the two ML algorithms were similar for the three domains, however each algorithm revealed various strengths and weaknesses. RS/RDR produced larger but more accurate KB from the cornerstone cases than Induct. The algorithm used by Induct uses probabilities and generalisation to determine rules for the most common classification and tries to find the attribute value pair that identifies cases with this classification. Once this rule has been added the cases are removed and new rules do not consider whether previously classified cases would now be misclassified. This problem is exacerbated by the way missing values are handled because a case may be treated as being covered by a rule because a value was missing when in fact if the value were known the case would not be covered and a new rule would be needed to cover it (a false positive). RS/RDR produces more specific rules that are generated from comparison of each case with every other case with the same conclusion. Only after evaluation of all cases are rules generated. Therefore, as confirmed by the figures in this study, the initial error rate with Induct is lower than with RST, but as more cases are used for training RST produces rules with less errors. Induct produces rules of around 2.5[6] clauses, which is bigger than the 1.3 clauses for manually built

[5] We only show the results from using KBS that had been randomized using a seed of 10. We have performed the same experiments on KBS generated from the 21822 Garvan cases and randomized using seeds of 10, 20, 30 and 40, with similar results.
[6] An average of 3.8 clauses per rule is reported in Catlett (1992) from Gaines (1989).

RDR but closer than for RS/RDR which averaged 3.3 conditions per rule. It is also interesting to note that when similarity indices between sets of rules were computed using a nearest neighbour algorithm it was found that the rules in the KBs produced using rough sets were closer to the Initial KB than the rules in the KBs produced using Induct.

In addition to the generation of reducts to remove redundancy, the significance coefficient can be treated as a measure of relative redundancy. If the attribute is dispensable then that attribute is redundant and not significant. When dealing with a set of rules one would expect that all attributes would contribute to at least one outcome so the measure determined is one of comparative significance. If discarding one attribute changes the classifications of many objects then it is more significant than another attribute which only alters the classification of a few objects when it is dropped. The computation of this coefficient may be beneficial if we wish to determine whether or not to drop an attrribute and to pinpoint the key features of a particular class. Two examples are given.

Example One
Significance Coefficients for rules concluding Volcanic (%VC000) or Plutonic (%PL000) are:
SignifCoef of attr 0 (GRAIN_SIZE) = 1.000000
SignifCoef of attr 3 (SILICA) = 0.600000

Example Two
Sig. Coeff. for rules concluding granodiorite %GR000, granite %GR001 and microgranite %GR002:
SignifCoef of attr 0 (GRAIN_SIZE) = 0.666667
SignifCoef of attr 2 (QUARTZ) = 0.666667

Figure 5: Significance coefficients produced by rough set theory for two different sets of mutually exclusive conclusions.

The first example, for the conclusions %VC000-Volcanic and %PL000-Plutonic, tells us that the attribute GRAIN_SIZE is very important in deciding whether a rock should be classified as volcanic or plutonic. In fact all rules for these conclusions used this attribute. However, in some situations, the attribute SILICA will also affect the conclusion. This interpretation is also confirmed by the work using FCA and can be seen in the labelled line diagram in figure 9. The second example, for the %GR family of conclusions, tells us that GRAIN_SIZE and QUARTZ are equally important attributes which help us to decide if a rock should be classified as %GR000-granodiorite, %GR001- granite or %GR002- microgranite. The significance coefficient μ, is calculated by determining the number of objects positively classified minus the number of objects positively classified once that attribute has been dropped, divided by the number of objects in the universe of discourse. Formula One returns the significance of *attr* in the set of attributes P in the active information system.

$$\mu_P\{attr\} = \text{card}\,(POS_P(R) - POS_{P-\{attr\}}(R))\,/\,\text{card}\,(U) \qquad [1]$$

where: R={all attributes);card = cardinal number of, POS = positive region, U = universe of objects.

Before going on to look at the other anomalies, it is useful to consider what benefits or drawbacks may result from this reorganisation process. Returning to the SISYPHUS III example, on the positive side, three conditions (attribute-value pairs) were removed and the number of concepts derived from the rules using formal concept analysis reduced from 78 to 56, signifying a 29% drop in rule-base complexity. By generalizing the rules, that is making them less specific by removing conditions, we should also develop more higher concepts than we get from rules designed just to classify one example, or rock in the case of the geology domain. On the possibly negative side, this removal of conditions may result in more unseen cases being misclassified compared to the original rule-base which contained more specific rules. The strategy of keeping all cases seen, mentioned before in the removal of redundancy, and using them for evaluation before and after is seen as a way of reducing the problem of creating rules that are too general or specific.

For the purposes of making the knowledge more perspicuous and useful for explanation, the decision of whether to minimise the rule-base depends on our goal. If the purpose of our KBS is to provide a tutoring system that gives a complete description of a rock then removal of *dispensable* conditions is not desirable. If our goal is to perform inferencing and to understand what are the key features for classifying a rock, then minimisation is appropriate. Minimisation addresses the issue of redundancy but may not pursue the validation goals of the system.

Further work will be conducted in the near future that looks at removing repetition from MCRDR KBS, since our focus has moved to MCRDR due to its superiority over single classification RDR. We plan to test the suitability of RST and the CUT95 Algorithm (Scheffer 1996) for this purpose. A feature also desirable in any approach to automatic reorganisation of RDR KBS, but not addressed in this work, is the output of a set of cases associated with each rule so that future system development can proceed as usual with manual KA. Now let us consider checks for consistency.

5.1.3 PHASE TWO: REMOVAL OF INCONSISTENCIES

The issue of ambivalence or inconsistency is addressed by computation of the degree of dependency κ which is simply the number of cases in the positive region divided by the number of all cases. For the decision algorithm PQ we can say that the set of attributes Q, the conclusion, depends in degree κ on the set of attributes P, the conditions, (see formula Two). If the degree of dependency is equal to one the knowledge base is consistent. If the degree of dependency is greater than zero but less than one we can determine what cases are outside the positive region and investigate why they are inconsistent.

$$\kappa = \text{card } POS_P(Q) \,/\, \text{card }(U) \qquad [2]$$

The 40 rules (objects) output from phase one (Section 4.1.2) were processed again to determine whether inconsistencies existed. Use of the dependency coefficient on all the rules in an MCRDR KBS is not appropriate because there are a number of conclusions valid for each case so that certain combinations of attribute-value pairs will result in multiple classifications, not just one, being assigned. When we determined κ for the whole SISYPHUS III KBS we got a dependency coefficient of 0 because the positive region was empty. It was therefore necessary to treat each classification individually and compute the dependency coefficient for each class to ensure that within each classification there was no inconsistency. However, it is desirable that rules which give mutually exclusive conclusions be verified for consistency. To achieve this in the SISYPHUS III KBS the rules for those conclusion known to be related but mutually exclusive were put into individual decision tables (files) and the dependency coefficient of each table computed. This was done for the conclusion groups: Plutonic (%PL000) and Volcanic (%VC000); Basic rock (%BS000) and acidic rock (%AC000); granodiorite %GR000, granite %GR001 and microgranite %GR002. Each group scored a dependency coefficient of 1 stating that for that type of decision the rules were consistent. A further indicator of the consistency of the knowledge is the "number of good decisions" reported by the rough set library after each classification. Accuracy was 100% for each individual classification and each classification group.

5.1.4 PHASE THREE: DETECTION OF DEFICIENCIES

The case driven nature of KA in RDR means that any value found in a case is valid and there is no restriction on or presupposition of what values or even what attributes are valid. However, some work has been done to detect what values in a case may indicate an abnormal value by (Edwards 1996). In his work on *Exceptions* valid value ranges are maintained by the system and any new values outside of the range are flagged and reported to the user as possibly erroneous. An immature KBS will give many warnings and high user involvement is expected at the beginning. The major frustration to the user occurs when a mature KBS is still giving false positives due to slight variations in values. To speed up this maturation process and reduce the notification of false invalid values the ranges can be preset to expected ranges. The drawback of this, however, is that this may be difficult and the person specifying the ranges must be a true expert familiar with the cases otherwise the ranges themselves may be invalid.

Deficiency or incomplete knowledge can be seen as the situation where no hypothesis can be inferred. The quality of classification can be tested in RST with the classification coefficient using the following formula:

$$\gamma_P(X) = \mathrm{card}(\underline{P}(X) \cup \underline{P}(-X)) / \mathrm{card}(U) \qquad [3]$$

This formula takes the number of objects in the union of the lower approximation of a set and its complement and divides this by the number of all objects in the domain and returns the quality of classification {X,-X} with respect to P in the active information system. This formula is checking whether all objects that are consistent can be classified. If we get a coefficient of 1 it means that all objects are consistent and can be classified. When we compute this value using the 40 SISYPHUS III reduced rules we get a classification coefficient of 1. This computation only provides a partial measure of the completeness of KBS.

There is another aspect of completeness that is more of a validation issue. This concerns how much of the domain is covered by the rules. At some point in time within a mature subdomain we may have seen cases that cover every combination of Attribute-Value (A-V) pairs. This can not be found through the detection of the above anomalies but some work done by Edwards (1996) on providing system *credentials* and subsequent work using simulated experts to perform this task (Compton et al 1996) does address this concern. The idea behind *credentials* is that if we gather statistics on how often each rule has fired correctly, how many corrections have been made to the rule and how recently modifications have been made we can get some idea of the rules stability. If we gather the same statistics by conclusion we can get a feel for the stability and coverage of the knowledge for the whole domain and for subdomains. If a KBS is seeing 100 cases a day and requiring 5 modifications then we can say that the KBS is 95% accurate and complete. This percentage can be finer-tuned by determining if certain parts of the KBS, which may cover a more recently introduced subdomain of the knowledge or a subdomain where knowledge is still emerging, are the parts that are being corrected. The more established subdomains may show 99%-100% completeness. In the PEIRS system there were a number of subdomains, including blood-gases, thyroid, liver functions, endocrines and catecholamines, that were added incrementally over the four years to make up the 2000+ rule-base. In the next section we look further at validation. *Credentials* can also be seen as an alternative solution to the explanation problem because we can use the system's credentials as a means of determining whether to accept the conclusion or not, just as we will seek and accept advice from experts that come highly recommended.

6. RDR and Validation

To further strengthen our approach to validation we have developed a technique which assists the user with KA in those situations where they are not as sure of the conclusion they should assign or the rule they should form. To this end we have been looking at providing KA in a critiquing mode so that when a user selects a certain conclusion it is evaluated against all other pathways (rules). If those rules are inconsistent with the current case the user is notified of the discrepancy. In Figure 6, again using the data from the SISYPHUS III experiments, we have attempted to add a rule that states that a particular rock, already classified as plutonic, should also be classified as volcanic. As previously mentioned, these conclusions are mutually exclusive and therefore it is inconsistent to assign both conclusions to the same case. This inconsistency has been detected by the system. The user is shown the rules in conflict and can then change their conclusion, select an existing rule to add an exception rule or add a new rule higher in the tree. Once the changes are made the case is run again and the user is shown all conclusions for that case. This should now show the new conclusions and any previous conclusions that were not being altered.

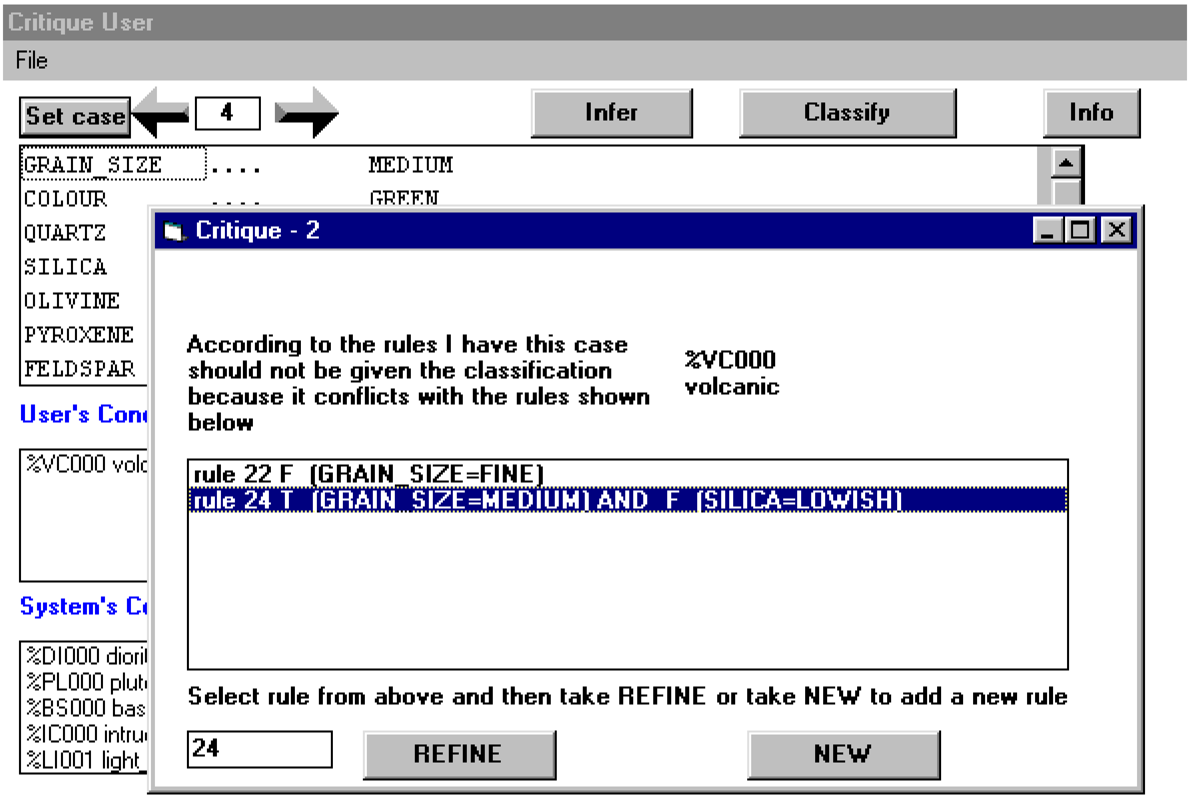


Figure 6: The critiquing screen in MCRDR/FCA, the user is shown inconsistent pathways

In additions to showing whether the conclusion is consistent with existing knowledge assistance is given to the user with selecting the rule conditions for the new rule. Using Formal Concept Analysis we are able to compare not only the primitive rule pathways, as we did for conclusions, but also the higher concepts that are implicit in the KBS. We first describe the mathematical foundation for FCA and then look at how we have applied it.

6.1 Formal Concept Analysis

Formal Concept Analysis, first developed by Wille (1982), is “based on the philosophical understanding of a concept as a unit of thought consisting of two parts: the extension and intension (comprehension); the extension covers all objects (entities) belonging to the concept while the intension comprises all attributes (or properties) valid for all those objects” (Wille 1992, p.493). The following description of FCA follows Wille (1982) which we apply to the SISYPHUS III KBS.

FCA, as does RDR, sees knowledge as existing and applying in a context (Compton and Jansen 1990, Wille 1996). FCA formalizes this notion by building crosstables known as a *formal context*. A formal context (K) has a set of objects G (for *Gegenstande* in German) and set of attributes M (for *Merkmale* in German) which are linked by a binary relation I which indicates that the object g (from the set G) has the attribute m (from the set M) and is defined as: K = (G,M,I). In our use of FCA we treat each rule pathway as an object. Since the I relation is binary, and an RDR KBS represents a many-valued context, we need to extend our notion of an attribute to cover a rule condition. To achieve this we simply treat the rule conditions, which are A-V pairs, as attributes. Thus in figure 7 we have the formal context K for the conclusions Plutonic - %PL000 and Volcanic - %VC000 with G = {21-%PL000, 23-%PL000, 25-%PL000, 22-%VC000, 24-%VC000}, where the object name is made up of the rule number and the conclusion, and M = {grain_size=course, 1=1,grain_size=medium, silica=intermediate, silica=very_high, grain_size=fine, silica=lowish}. The crosses show where the relation I exists, thus I = {(21-%PL000, grain_size=course), (21-%PL000, 1=1), (23-%PL000, grain_size=medium), (23-%PL000, silica=intermediate), (23-%PL000, 1=1),…,(24-%VC000, silica=lowish)}. This is similar to the technique known as *conceptual scaling* (Ganter and Wille 1989) which has been used to interpret a many-valued context into a (binary) formal context. A many-valued context is a quadruple (G,M,W,I) where I is a ternary relation between the set of objects G, the set of attributes M and the set of attribute values W (merkmalsWerte in German). Conceptual scaling basically treats each attribute as a separate formal context with the values as attributes associated with each of the original objects. A scale is chosen, such as a nominal scale (=) or an ordinal scale (≥), to order these attributes. From the many contexts, one for each attribute, the concepts are derived.

	grain_size =coarse	1=1[7]	*grain_size =medium*	silica= intermediate	silica= very_high	*grain_size =fine*	silica= lowish
21-%PL000	X	X					
23-%PL000		X	X	X			
25-%PL000		X	X		X		
22-%VC000		X				X	
24-%VC000		X	X				X

Figure 7: Formal Context for the conclusions Plutonic (%PL000) and Volcanic (%VC000). Column One gives the object - which in our use of FCA corresponds to a rule pathways. We identified each object uniquely by its rule number and the conclusion it gives. The columns are the attributes which correspond to the conditions in our rules. An ‘X’ signifies that a particular rule has the corresponding condition.

A formal concept is a pair (X,Y) where X is the *extent*, the set of objects, and Y is the *intent*, the set of attributes, for the concept. The derivation operators in formulas 4 and 5:

$$X \subseteq G: \; X \mapsto X' := \{m \in M \mid gIm \text{ for all } g \in X\} \quad [4]$$

$$Y \subseteq M: \; Y \mapsto Y' := \{g \in G \mid gIm \text{ for all } m \in Y\} \quad [5]$$

[7] 1=1 is our default condition in Rule 0 which is the first rule which provides the default conclusion assigned to cases that do not satisfy any of the other rules in the KBS. See Rule 0 in Figure 1 for an example MCRDR KBS. We could remove this condition from the formal context if deemed appropriate.

are used to construct all formal concepts of a formal context, by finding the pairs (X″,X′) and (Y′,Y″). We can obtain all extents X′ by determining all row-intents {g}′ with g ∈ G and then finding all their intersections. Another approach it to find all intents Y′ by determining all column-extents {m}′ with m ∈ M and then finding all their intersection. This is specified in formulas 6 and 7 as:

$$X' = \bigcap_{g \in X} \{g\}' \quad [6] \qquad Y' = \bigcap_{m \in Y} \{m\}' \quad [7]$$

Less formally, we take the set of objects, G, to form the initial extent X which also represents our largest concept. We then process each attribute sequentially in the set M, finding the intersections of the extent for that attribute with all previous extents. Once the extents have been found for all attributes, the intents X′ for each extent X may be found by taking the intersection of the intents for each object within the set. Thereby we determine all formal concepts of the context K by finding the pairs (X,X′). The formal concepts for our example are shown in the concept matrix in Figure 8. All screens shown are our implementation, called MCRDR/FCA.

The next step is to find the subconcept-superconcept relation between concepts so that they may be ordered. We can use the subsumption relation ≤ on the set of all concepts formed such that $(X_1,Y_1) \leq (X_2,Y_2)$ iff $X_1 \subseteq X_2$. For a family (Xi,Yi) of formal concepts of K the greatest subconcept, the join, and the smallest superconcept, the meet, are respectively given by formulas 8 and 9:

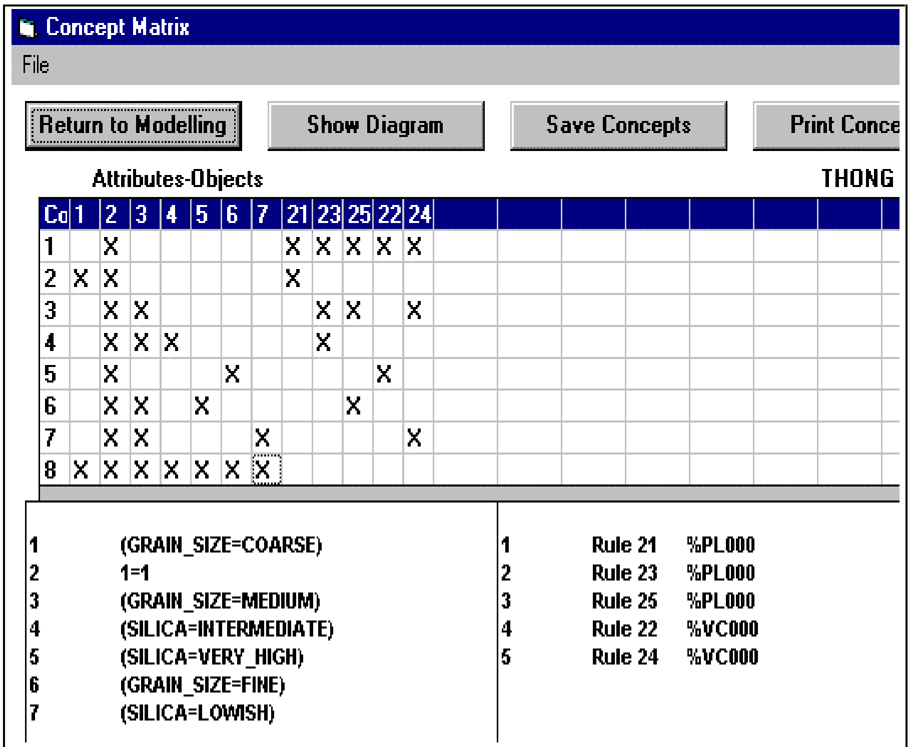
$$\bigvee_{i \in I} (X_i,B_i) := \Big(\big(\bigcup_{i \in I} A_i\big)'', \bigcap_{i \in I} B_i \Big) \quad [8] \qquad \bigwedge_{i \in I} (X_i,B_i) := \Big(\bigcap_{i \in I} A_i, \big(\bigcup_{i \in I} B_i\big)'' \Big) \quad [9]$$

From Lattice Theory, we are able to form a complete lattice, called a concept lattice and denoted B(K), with the ordered concept set. The concept lattice provides "hierarchical conceptual clustering of the objects (via the extents) …. and a representation of all implications between the attributes (via its intents)" [Wille 1992, p.497]. The line diagram, also known as a *Hasse* diagram, is shown in Figure 9. Each concept has various intents and extents associated with it and is shown as a circle. The lines represent the sub/superconcept relations. For easier reading the labeling is reduced as in Figure 9. All extents are reached by descending paths from the concept β and all intents of a concept β are reached by ascending paths from β.

6.2 Validating New Concepts Through Comparison with Old Concepts Using FCA

The work by Boswell, Craw and Rowe (1997) mentioned earlier looks at using rule traces to allow KRUST to propose new knowledge. The approach in this section is somewhat different as we look at a way of re-representing the rule pathways in an MCRDR KBS in order to determine the impact of new knowledge on existing knowledge. It was out goal to understand the underlying relationships and models inherent in the RDR rules to assist us with the validation of new rules. The following discussion refers to our MCRDR/FCA implementation. To give a fuller understanding of the approach we explain and show the concepts derived and then describe how this can assist validation.

As described in the previous section, finding concepts using FCA begins with the definition of a formal context. When using an MCRDR KBS the first step is conversion of the exceptions into a flat structure of pathways that corresponds to a formal context. Pathways were found by reading each rule and picking up the conditions from the parent rule until the top node with the default rule was reached. From this flattened KBS the user chooses either the whole KB or a more narrow focus of attention from which to derive a formal context. When the whole KB is chosen the rules and rule conditions form the extents and intents, respectively. If our goal is to display the concepts as a line diagram then it is advisable to reduce the size of the formal context by selecting a focus of attention. There are currently thirteen possible ways that a context can be derived. It is also possible to combine contexts. The two most common methods are to create a context based on a selected rule or conclusion. If a conclusion is chosen, all rules using that conclusion are selected and added as objects to the set G, forming the extents of the context. As each extent is added the conditions of the rules are added to the set M of attributes to form the intents of the context, first checking to see if any attributes have already been added by previous rules. Where the relation I held, that is object g had attribute m, a cross was marked in the appropriate row and column. In Figures 8 and 9 we are shown the concept matrix and lattice, respectively, when the conclusions Plutonic-%PL000 and Volcanic-%VC000 were chosen.



Co	1	2	3	4	5	6	7	21	23	25	22	24
1		X						X	X	X	X	X
2	X	X						X				
3		X	X						X	X		X
4		X	X	X					X			
5		X				X					X	
6		X	X		X					X		
7		X	X				X					X
8	X	X	X	X	X	X	X					

1	(GRAIN_SIZE=COARSE)
2	1=1
3	(GRAIN_SIZE=MEDIUM)
4	(SILICA=INTERMEDIATE)
5	(SILICA=VERY_HIGH)
6	(GRAIN_SIZE=FINE)
7	(SILICA=LOWISH)

1	Rule 21	%PL000
2	Rule 23	%PL000
3	Rule 25	%PL000
4	Rule 22	%VC000
5	Rule 24	%VC000

Figure 8: The Concept Matrix Screen from MCRDR/FCA. Here the user is shown the concepts for the context generated from the rules which conclude Plutonic-%PL000 or Volcanic-%VC000 for the SISYPHUS III geology domain. Each row is a concept. The columns show the attributes first -numbered 1-7 followed by the objects which are labeled by the rule number they represent. The labeling has been reduced to numbers so that the patterns that emerge may be seen. Two pop-up windows are shown at the bottom. The left window gives the attribute names and the right window gives the object names.

If the user chooses a particular rule then that rule is added as the first object with the rule conditions as the initial *intent*. Every condition in each rule in the flattened RDR KBS is searched for a match on the initial set of attributes. If a match is found, that rule is added to the extension and all new attributes (conditions) found in the matching rule are also added to the intent. Our approach is similar to that proposed by Ganter (1988) where the context is shortened to find sub-contexts and subrelations. The decomposition of a concept lattice into smaller parts is a strategy that has previously been found useful (Wille 1989a). For our purpose of finding how our proposed rule fits in with existing concepts it is advisable to select the whole KB since we are not worried about displaying too many concepts and do not wish to miss any relevant concepts.

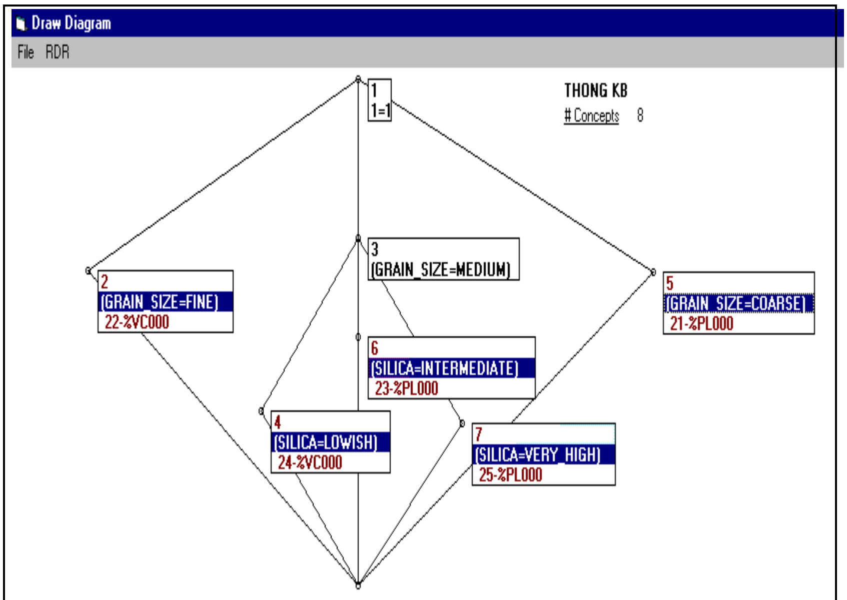


Figure 9: The Labeled Line Diagram for the PLUTONIC (%PLOOO) and VOLCANIC (%VCOOO) conclusions using MCRDR/FCA for Windows .We can see if GRAIN-SIZE = COURSE a rock is plutonic (%PL000) and if GRAIN-SIZE = FINE a rock is VOLCANIC (%VC000). However, when GRAIN-SIZE = MEDIUM then if SILICA = LOWISH it is a volcanic rock otherwise if SILICA = VERY-HIGH or INTERMEDIATE it is a plutonic rock. The line diagram has shown us what attribute-value pairs are the critical ones for these conclusions.

Having generated a formal context we can then construct all formal concepts of the formal context, using the process described in section 6.1. From our set of ordered concepts we compute the predecessors and successors of each concept so that we can draw the *Hasse* diagram. Predecessors are determined by finding the greatest subconcept, formula 8, of the intents for each concept. Successors are determined by finding the smallest superconcept, formula 9, of the intents. The successor list is used to identify concepts higher in the diagram, the parents, and the predecessor list identifies concepts lower in the diagram, the children.

There are many ways that a line diagram can be drawn. It may be desirable to provide a number of different layouts because concepts can be viewed and examined in different ways depending on their purpose and meaning (Wille 1992). In MCRDR/FCA the user has the ability to move a node anywhere they like providing the node is not moved higher than any of its parents or lower than any of its children. The algorithm used for ordering will affect the appearance of the concept matrix, see Figure 8, but does not affect the calculation of predecessors and successors or the graph layout. Appropriate ordering of concepts in the two-dimensional matrix can be difficult as a given concept may be a subconcept of different superconcepts. This can be seen in the concept matrix in Figure 8 where we can see that concept number 3 is a superconcept of concepts number 4, 6 and 7. This is more evident in the line-diagram in Figure 9.

So far we have used MCRDR/FCA in three different domains. The first was a 60-rule Blood Gases KBS, known as 105, that had been developed from the cornerstone cases associated with the 2000+ PEIRS rules. The second domain was known as LOTUS and concerned the adaptation and management of the *Lotus Uliginosis cv* Grasslands Maku for pastures in the Australian state of New South Wales (Hochman et al 1996). The knowledge was recorded into four KBS by four independent agricultural advisors. We used the concept matrices and line diagrams to compare the conceptual models of the advisors. This technique was seen as a useful way to identify and reconcile any differences as well as identify the main concepts associated with this domain. The third domain is the SISYPHUS III geology domain we have been using in this paper.

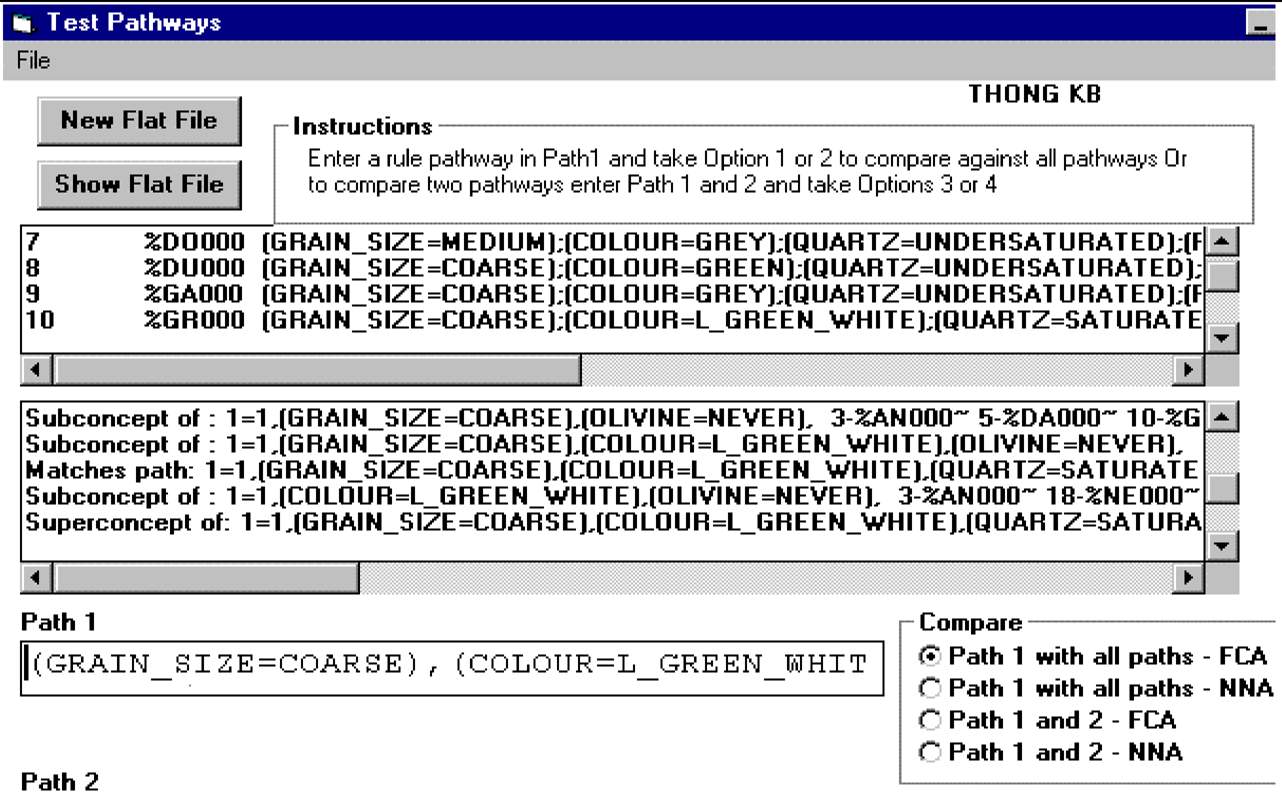


Figure 10: The user can compare the proposed rule, shown in the Path 1 textbox, with existing concepts using FCA or a nearest neighbour algorithm. The user also has the option of showing the whole existing flat file (in the top list box) from which they can select a particular path they can compare against by putting that pathway in the Path 2 textbox (not shown). The second listbox shows the user how whether the rule is a match, sub or superconcept if one of the FCA options are taken or a score between 0-1 for each rule pathway using a nearest neighbour algorithm.

The discussion thus far shows how these concepts are found and can be displayed but to assist the user with KA and validation we have added another screen. When the user forms a new rule they can click a new EVALUATE rule button which takes them to the screen shown in Figure 10. The new rule that they propose to add is put into the Path 1 textbox and they can choose to compare the proposed pathway against the concepts found using FCA or use a nearest neighbour algorithm that provides a score between 0-1. This score is based on the number of pattern matches and number of conditions in each pathway compared to the pathway under consideration. When the user chooses to compare the proposed rule using FCA they are presented with a listbox of all the concepts found using FCA that are matches, sub or superconcepts. For this purpose we are only using the intensional definition of the concepts derived using Wille'e technique since the extensional definition is too restrictive. This is because an intensional definition implies an extensional definition but the converse is possibly but not necessarily true (Zalta 1988). If the user selects the nearest neighbour algorithm (NNA) option they are shown all pathways and scores. The purpose of showing the user this information is to give them an understanding of how the new rule fits in with the existing knowledge. If the new rule is identified with concepts that seem inappropriate this is a warning to the user that the knowledge in the new rule or an existing rule is incorrect, the decision of whether and in what way to amend the rule is left to the user.

We can see from the examples that the ability to find and compare concepts in our knowledge base is useful for validation purposes. Such an approach goes beyond verification and attempts to demonstrate whether the knowledge is an accurate representation of the expert's knowledge. As a next step we want to improve the user friendliness of our screens and test our approach using real experts, seeing if giving the user more understanding of the knowledge already captured assists them when adding new rules.

The abstraction hierarchy developed by FCA can be seen as a domain ontology. The motivation for the development of an ontology and the type of ontology provided by the FCA line diagrams is in keeping with the general "aim of ontology research [which] is to explicitly represent the meaning of concepts and the relation among them" and the notion that an "ontology, in general, is an agreement between users and systems" (Ikeda et al 1996). Ontological commitment can be a problem when there is no direct coupling between the KBS and the ontology (Ikeda et al 1996, Waterson and Preece 1997). Obtaining this commitment may require mappings between KR and translation to a common set of terminology (Waterson and Preece 1997). A key benefit of using FCA to automatically generate an ontology is that we are able to say that our KBS commits to the ontology.

7 Other Work and Conclusions

We have begun looking at the repetition problem for MCRDR KBS and a proposal has been developed for this problem but is yet to be tested. We have performed evaluations of our line diagrams with novices and experts to determine the usefulness of the representation for explanation and learning about a domain. The results have been very promising particularly from the experts. It appears that the model developed by FCA provides a useful basis for discussion and learning about a domain. This learning includes the identification of erroneous models (KBS needs fixing) or interpretations by the user (student needs to revise their hypothesis). Another use of the MCRDR/FCA tool has been the development of a Requirements Engineering (RE) framework (Richards and Menzies 1998) which detects conflict between multiple viewpoints and offers a range of negotiation operators to manage the conflict. Included in the framework is an evaluation method for determining that the distance between viewpoints is decreasing.

The RDR philosophy of capturing knowledge with minimal analysis of the domain is supported by the system we have described in this paper. By automatically generating the concepts after KA we are able to gain the benefits that models provide without the restrictions and problems that developing those models imposes. There remains a number of parts of the system that need to be finer tuned. As mentioned in section 6.1, the formulation of concept lattices from many-valued contexts requires their interpretation into a binary formal context. While we have taken a simple approach by treating a rule condition as an attribute, currently a rule that should be part of the context for a selected focus of attention may be missed if the condition does not match on a conclusion or attribute already selected. The use of different *conceptual scales* (Ganter and Wille 1989) may provide a solution and needs further investigation. As one option we have looked at prespecifying boundary categories that are used to pick up attribute values that border with the value being selected. We have already done some work using a distance-weighted nearest neighbour algorithm to assign a score to conditions to find if and to what extent other conditions are related which could possibly be used to determine which rules should be added to a context. The issue of multi-contexts, the appropriate views for selecting contexts, the best diagrams layouts and other user interface issues will be domain dependent and may not be properly answered until they are guided by an application.

The importance of the expert in the loop has always been a key feature of RDR systems. This is also apparent in the recent verification research reported in Section 5 where human intervention is required to determine whether the anomalies detected by the system actually require fixing and those fixes will include a mixture of automatic and manual procedures. The validation described in

Section 6 involves an even more cooperative effort where the computer becomes a second expert with whom the human expert can confer.

This paper has focused on the use of RST for verification and FCA for validation for RDR KBS but both techniques have been applied to other rule-based representations (Richards, Gambetta and Compton 1996, Richards 1998). The approaches are applicable to any knowledge representation that can be mapped into a decision table (for RST) or binary crosstable (for FCA). This process is very simple when RDR are used as input as there are no intermediate, user interface or control-type rules to convert. The retrospective development of an abstraction hierarchy using FCA is particularly useful for RDR since such models are not developed as part of the KA process. The technique can also be beneficial for validating that the KBS built using other KA approaches agrees with the initial domain models developed.

In conclusion, the automatic V&V provided by the RDR exception structure and the use of cases has been reported by a range of authors and demonstrated in the successful PEIRS application. Our recent extensions which include ideas from rough set theory for verification and formal concept analysis for validation seem promising but have yet to be deployed for a real application where, no doubt, certain modifications may be required.

Acknowledgements
RDR research is supported by Australian Research Council grants. The authors would like to thank Bob Colomb, Jacek Sienkiewicz, Zdzislaw Pawlak, Frank Vogt and Rudolf Wille for their assistance and encouragement with this work.

References

Barr, Valerie (1997) Applications of Rule-Base Coverage Measure to Expert System Evaluation In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Applications of Artificial Intelligence Conference* AAAI Press/ MIT Press, Cambridge, Massachusetts, 411-416.

Boswell, Robin, Craw, Susan and Rowe, Ray (1997) Knowledge Refinement for a Design System, In Enric Plaza and Richard Benjamins (Eds), *Knowledge Acquisition, Modeling and Management*. 10th European Workshop, EKAW'97, Lecture Notes in Artificial Intelligence 1319, Springer-Verlag, 49-64.

Catlett, J. (1992) Ripple-Down-Rules as a Mediating Representation in Interactive Induction *Proceedings of the Second Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop*, Kobe, Japan, Nov 9-13 1992, 155-170.

Chander, P.G., Shinghal, R. and Radhakrishnan, T. (1997) Performance Assessment and Incremental Evaluation of Rule-based Systems In Plant, R. and Antoniou, G. (eds) *Proceedings of AAAI'97 Verification and Validation of Knowledge-Based Systems Workshop,* July 28, 1997, Providence, Rhode Island, Technical Report WS-97-01, AAAI Press, Menlo Park, California, 40-46.

Chandrasekaran, B. (1986) Generic Tasks in Knowledge-Based Reasoning: High Level Building Blocks for Expert System Design IEEE Expert, Fall 1986, 23-30.

Chellen, Vijaletchmee (1995) Induct/RDR and Knowledge Base Compaction *BE Computer Engineering Honours Thesis*, University of New South Wales.

Clancey, W.J., (1991) The Frame of Reference Problem in the Design of Intelligent Machines In K. VanLehn, ed. *Architectures for Intelligence,* Erlbaum, Hillsdale.

Colomb, Robert.M. (1993) Decision Tables, Decision Trees and Cases: Propositional Knowledge-Based Systems *Technical Report No. 266* Key Centre for Software Technology, Department of Compter Science, The University of Queensland, Australia.

Colomb, R.M. and Sienkiewicz, J. (1995) Analysis of Redundancy in Expert Systems Case Data Xin Yao (ed) *Proceeding of the Eighth Joint Conference on Artificial Intelligence AI'95* 13-17 November 1995, Canberra, World Scientific, Singapore.

Compton, P. and Jansen, R., (1990) A Philosophical Basis for Knowledge Acquisition. *Knowledge Acquisition* 2:2

Compton, P., Edwards, G., Kang, B., Lazarus, L., Malor, R., Preston, P., and Srinivasan, A. (1992). Ripple down rules: Turning knowledge acquisition into knowledge maintenance *Artificial knowledge in Medicine 4* pp:463-475.

Compton, P., Kang, B., Preston, P. and Mulholland, M., (1993), Knowledge Acquisition without Analysis, *European Knowledge Acquisition Workshop,* Springer Verlag, 277-290.

Compton, P., Preston, P. and Kang, B. (1995) The Use of Simulated Experts in Evaluating Knowledge Acquisition, *Proceedings 9th Banff Knowledge Acquisition for Knowledge Based Systems Workshop* Banff. Feb 26 - March 3 1995.

Compton, P, Preston, P., Edwards, G, and Kang, B. (1996) Knowledge Based Systems that have some idea of Their Limits *Proceedings 10th Banff Knowledge Acquisition for Knowledge Based Systems Workshop* Banff. November 1996.
Cragun, B.J. and Steudel, H.J. (1987) A Decision-Table-Based Processor for Checking Completeness and Consistency in Rule-Based Expert Systems *Int. Journal of Man-Machine Studies* 26(5):633-648.
Dimopoulos, Yannis and Kakas, Antonis, (1995) Learning Non-Monotonic logic Programs: Learning Exceptions. In Nada Lavrac and Stefan Wrobel (eds) *Machine Learning: Proceedings of the European Conference on Machine Learning ECML-95* Lecture Notes in Artificial Intelligence 912, Springer Verlag, Berlin, 122-137.
Edwards, G., Compton, P., Malor, R, Srinivasan, A. and Lazarus, L. (1993) PEIRS: a Pathologist Maintained Expert System for the Interpretation of Chemical Pathology Reports *Pathology 25*: 27-34.
Edwards, Glenn (1996) Reflective Expert Systems in Clinical Pathology *MD Thesis*, University of New South Wales.
Fibak, J., Slowinski, K. and Slowinki, R. (1986) Rough Sets Based Decision Algoritm for Treatment of Duodenal Ulcer by *HSV Proceedings of 6th Internal Workshop on Expert Systems and their Applications* Avignon, France, pp:587-599.
Gaines, B. R. (1989) Knowledge Acquisition: the Continuum Linking Machine Learning and Expertise Transfer *Proceedings of the 3rd European Workshop on Knowledge Acquisition for Knowledge-Based Systems* Paris, 90-101.
Gaines, B.R., (1991) Induction and Visualization of Rules with Exceptions In J. Boose & B. Gaines, Eds, *Proceedings of the 6th Banff AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop.* Vol 1:7.1-7.17, Banff, Canada:
Gaines, B. R. and Shaw, M.L.G. (1989) Comparing the Conceptual Systems of Experts *The 11th International Joint Conference on Artificial Intelligence* :633-638.
Ganter, B. (1988) Composition and Decomposition of Data *In Classification and Related Methods of Data Analysis* (Ed. H. Bock) pp:561-566, North-Holland, Amsterdam.
Ganter, B. and Wille, R. (1989) Conceptual Scaling In *Applications of Combinatorics and Graph Theory to the Biological Sciences* (Ed. F. Roberts) pp:139-167, Springer, New York.
Gonzalez, A.J. and Dankel, D.D. (1993) *The Engineering of Knowledge-Based System* Prentice Hall.
Grosof, Benjamin N. (1997) Courteous Logic Program: Prioritized Conflict Handling for Rules. *IBM Research Report* RC 20836(92273) Available from:
http://www.research.ibm.com/people/g/grosof.
Grossner, C., Gokulchander, P. Preece, A., and Radhakrishnan, T. (1996) Revealing the Structure of Rule-Based Systems *International Journal of Expert Systems: Research and Applications* 9(2):255-278.
Guha, T.V., and Lenat, D.B. (1990) CYC:A Mid-Term Report *AI Magazine 11*(3):32-59
Helmbold, D., Sloan, R. and Warmuth, M.K. (1989) Learning Nested Differences of Intersection-Closed Concept Classes *Proceedings of the Workshop on Computational Learning Theory*, 41-56.
Hochman, Z., Compton, P., Blumenthal, M. and Preston, P. (1996) Ripple-down rules: a potential tool for documenting agricultural knowledge as it emerges. p313-316. In *Proc. 8th Aust. Agronomy Conf.*, Toowoomba 1996.
Ikeda, Mitsuru, Seta, Kazuhisa, Kakusho, Osamu and Mizoguchi, Riichiro (1996) An Environment for Building Conceptual Model of Problem Solving *Proceedings of Pacific Knowledge Acquisition Workshop PKAW'96*, October 23-25 1996, Coogee, Australia, 210-225.
Kang, B., Compton, P. and Preston, P (1995) Multiple Classification Ripple Down Rules: Evaluation and Possibilities *Proceedings 9th Banff Knowledge Acquisition for Knowledge Based Systems Workshop* Banff. Feb 26 - March 3 1995, Vol 1: 17.1-17.20.
Kang, B., (1996*) Validating Knowledge Acquisition: Multiple Classification Ripple Down Rules* PhD Thesis, School of Computer Science and Engineering, University of NSW, Australia.
Kang, B., Gambetta, W. and Compton, P. (1996) Verification and Validation with Ripple-Down Rules *Int. J. Human-Computer Studies* 44 :257-269
Kivinen, J., Mannila, H. and Ukkonen, E. (1993) Learning Rules with Local Exceptions *Proceedings of the European Conference on Computational Learning Theory COLT'93.*
Kolodner, Janet (1993) *Case-Based Reasoning* Morgan Kaufman Publishers Inc., San Mateo, CA.
Kowalski, R. and Sadri, F. (1991) Logic Programs with exceptions *New Generation Computing* 9:387-400.
Krysinski, J. (1990) Rough Sets Approach to Analysis of Relationship Between Structure and Activity of Quaternary Imidazolium Compounds*. ArzeneimittelForschung Drug Research, 40*, pp:795-799.
Li, X. (1991) What's so bad about rule-based programming ? *IEEE Software*, September 1991, 103-105.
Mansuri, Y., Kim, J.G., Compton, P. and Sammut, C. (1991*).* A comparison of a manual knowledge acquisition method and an inductive learning method *Australian Workshop on Knowledge Acquisition for Knowledge Based Systems*, Pokolbin (1991) pp:114-132.
McDermott, J. (1988) Preliminary Steps Toward a Taxonomy of Problem-Solving Methods *Automating Knowledge Acquisition for Expert Systems* Marcus, S (ed.) Kluwer Academic Publishers, pp: 225-256.
Menzies, T.J. and Compton, P. (1995) The (Extensive) Implications of Evaluation on theDevelopment of Knowledge-Based Systems in *Proceedings 9th Banff Knowledge Acquisition for Knowledge Based Systems Workshop* Banff. Feb 26 - March 3 1995,.

Meseguer, P. (1992) Incremental Verification of Rule-based Expert Systems In Neumann, B., ed. *10th European Conference on Artificial Intelligence*, 829-834.
Murphy, Patrick M. and Pazzani, Michael J. (1994) Revision of Production System Rule-Bases In W.W. Cohen and H. Hirsch, (eds.) *Machine Learning: Proceedings of the Eleventh International Conference,* New Brunswick, NJ, Morgan Kaufmann, 199-207.
Newell, A. (1982) The Knowledge Level *Artificial Intelligence* 18:87-127.
Nguyen, T.A., Perkins, W.A., Laffey, T.J. and Pecora, D. (1987) Knowledge Base Verification *AI Magazine* Summer 1987, 69-75.
O'Keefe, R.M., Balci, O. and Smith, E.P. (1987) Validating Expert System Performance *IEEE Expert,* 81-89.
O'Keefe, R.M. and O'Leary, E.E. (1993) Expert System Verification and Validation: A Survey and Tutorial Artificial Intelligence Review 66 273-309.
Patil, R. S., Fikes, R. E., Patel-Schneider, P. F., McKay, D., Finin, T., Gruber, T. R. and Neches, R., (1992) The DARPA Knowledge Sharing Effort: Progress Report In C. Rich, B. Nebel and Swartout, W., *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference* Cambridge, MA, Morgan Kaufman.
Pawlak, Zdzislaw (1982) Rough *Sets International Journal of Information and Computer Sciences, 11,* pp:341-356.
Pawlak, Zdzislaw (1991) *Rough Sets: Theoretical Aspects of Reasoning about Data* Kluwer Academic Publishers Dordrecht.
Pirlein, T and Studer, R., (1994) *KARO*: An Integrated Environment for Reusing Ontologies *A Future for Knowledge Acquisition EKAW'94* Springer Verlag, 200-225.
Preece, A.D., Shinghal, R and Batarekh, A. (1992) Principles and Practice in Verifying Rule-Based Systems *The Knowledge Engineering Review 7*:115-141.
Puerta, A. R, Egar, J.W., Tu, S.W. and Musen, M.A. (1992) A Multiple Method Knowledge Acquisition Shell for Automatic Generation of Knowledge Acquisition Tools *Knowledge Acquisition 4*(2).
Quinlan, J. Ross (1992) *C4.5 Programs for Machine Learning* Morgan Kaufman Publishers Inc.
Richards, D. (1998) An Evaluation of the Formal Concept Analysis Line Diagram In Grigoris Antoniou (ed.) *Poster Proceedings of Eleventh Australian Joint Artificial Intelligence Conference AI'98* 13-17 July 1998, Griffith University, Nathan Campus, Brisbane, Australia, (to appear).
Richards, D., Chellen, V. and Compton, P (1996) The Reuse of Ripple Down Rule Knowledge Bases: Using Machine Learning to Remove Repetition In Compton, P., Mizoguchi, R., Motoda, H. and Menzies, T. (eds) *Proceedings of Pacific Knowledge Acquisition Workshop PKAW'96*, October 23-25 1996, Coogee, Australia.
Richards, D ., Gambetta, W. and Compton, C (1996) Using Rough Set Theory to Verify Production Rules and Support Reuse In Antoniou, G. (ed) *Proceedings of the Verification, Validation and Refinement of KBS Workshop, PRICAI'96* 26-30 August 1996, Cairns, Australia, Griffith University, 57-66.
Richards, D. and Menzies, T. (1998) Extending the SISYPHUS III Experiment from a Knowledge Engineering to a Requirements Engineering Task In Gaines, B. and Musen, M. (eds) *11th Workshop on Knowledge Acquisition, Modeling and Management,* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:SIS-6.
Rivest, R.L. (1987) Learning Decision Lists *Machine Learning* 2(2):229-246.
Rosenfield, I., (1988) *The Invention of Memory: A New View of the Brain*, Basic Books, New York.
Santos, Eugene Jr., Gleason, Howard T. and Banks, Sheila B. (1997) BVAL: Probabilistic Knowledge-Base Validation In Plant, R. and Antoniou, G. (eds) *Proceedings of AAAI'97 Verification and Validation of Knowledge-Based Systems Workshop*, July 28, 1997, Providence, Rhode Island, Technical Report WS-97-01, AAAI Press, Menlo Park, California, 12-22.
Scheffer, T. (1996) Algebraic Foundation and Improved Methods of Induction of Ripple Down Rules *Proceedings of Pacific Knowledge Acquisition Workshop PKAW'96*, October 23-25 1996, Coogee, Australia.
Schreiber, G., Weilinga, B. and Breuker, J. (1993) KADS: A Principled Approach to Knowledge-Based System Development *Knowledge-Based Systems* London, England, Academic Press.
Shadbolt, N., (1996)URL:http://www.psyc.nott.ac.uk/aigr/research/ka/SisIII
Shiraz, G.M. and Sammut, C. (1998) Acquiring Control Knowledge from Examples Using Ripple-down ules and Machine learning, *11th Workshop on Knowledge Acquisition, Modeling and Management,* Banff, Canada, SRDG Publications, Departments of Computer Science, University of Calgary, Calgary, Canada, Vol 1:KAT-5.
Slowinski, R., Slowinski, K. and Stefanowski, J. (1988) Rough Set Approach to Analysis of Data from Peritoneal Lavage in Acute Pancreatitis *Medical Information, 13*:143-159.
Soloway, E, Bachant, J. and Jensen, K. (1987) Assessing the Maintainability of XCON-in-RIME: Coping with Problems of a very Large Rule Base *Proceedings of the Sixth International Conference on Artificial Intelligence* Vol 2:824-829, Seattle, WA: Morgan Kaufman.
Steels, L. (1993) The Componential Framework and Its Role in Reusability In David, J.M., Krivine, J.-P. and Simmons, R., editors *Second Generation Expert Systems* pp: 273-298. Springer, Berlin.

Suwa, M., Scott, A. and Shortliffe, E. (1982) An Approach to Verifying Completeness and Consistency in a Rule-Based Expert System *AI Magazine* 3(4):16-21.
Vere, S. A. (1980) Multilevel Counterfactuals for Generalizations of Relational Concepts and Productions *Artificial Intelligence* 14:139-164.
Waterson, Andrew and Preece, Alun (1997) Knowledge Reuse and Knowledge Validation In Plant, R. and Antoniou, G. (eds) *Proceedings of AAAI'97 Verification and Validation of Knowledge-Based Systems Workshop*, July 28, 1997, Providence, Rhode Island, Technical Report WS-97-01, AAAI Press, Menlo Park, California, 33-39.
Webb, G. and Wells, J. (1995) Recent Progress in Machine-Expert Collaboration for Knowledge Acquisition *Proceedings of Eighth Australian Joint Conference on Artificial Intelligence AI'95***,** Ed X. Yao, 13-17 November 1995, Canberra, World Scientific, 91-298.
Wille, R. (1982) Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts In *Ordered Sets* (Ed. Rival) pp:445-470, Reidel, Dordrecht, Boston.
Wille, R. (1989a) Lattices in Data Analysis: How to Draw them with a Computer In *Algorithms and Order* (Ed. I. Rival) pp:33-58, Kluwer, Dordrecht, Boston.
Wille, R. (1989b) Knowledge Acquisition by Methods of Formal Concept Analysis *In Data Analysis, Learning Symbolic and Numeric Knowledge* (Ed. E. Diday) pp:365-380, Nova Science Pub., New York.
Wille, R. (1992) Concept Lattices and Conceptual Knowledge Systems *Computers Math. Applic. (23)*6-9:493-515.
Wille, R. (1996) Conceptual Structures of Multicontexts *Conceptual Structures: Knowledge Representation as Interlingua* (Eds. P.Eklund, G. Ellis and G. Mann) pp:23-39, Springer.
Zalta, E.N. (1988) *Intensional Logic and the Metaphysics of Intentionality*, Cambridge, Massachusetts, MIT Press.