A Semantic Approach Towards Software Engineering of Knowledge Bases

Mala Mehrotra

Pragati Synergetic Research, Inc.
145 Stonelake Court
Yorktown VA. 23693
mm@infi.net

Abstract

Due to the critical nature of knowledge base applications in important intelligence-based environments, much more stringent standards have to be imposed now on their ability to provide reliable decisions in an accurate manner. It is our contention that in order to build reliable knowledge-based systems, it is important that the knowledge in the system be suitably abstracted, structured, and otherwise clustered in a manner which facilitates its understanding, verification, validation, maintenance, management and testing. The MVP-CA methodology addresses partitioning rule-based systems into a number of meaningful units before attempting the above activities. Pragati's Multi-ViewPoint-Clustering Analysis (MVP-CA) tool provides such a framework for clustering large, homogeneous knowledge-based systems from multiple perspectives. It is a semi-automated tool allowing the user to focus attention on different aspects of the problem, thus providing a valuable aid for comprehension, maintenance, integration and evolution of knowledge-based systems.

Motivation

With the advent of intelligent information systems, knowledge bases are being widely applied as intelligent information specialists both for civilian and military applications. Due to the critical nature of these applications, much more stringent standards have to be imposed now on their ability to provide reliable decisions in a timely and accurate manner. Most of the tools available commercially, try to apply conventional verification and validation (V&V) approaches to knowledge-based systems, instead of leveraging off of the fact that knowledge-based systems have a declarative style of programming. Therefore there has been very limited success in assuring the reliability of such systems. While using knowledge-based programming techniques, one is much closer to the domain knowledge of the problem than with procedural languages. The control aspects of the problem are abstracted away into the inference engine (or alternatively, the control rules are explicitly declared). Hence, **validating the knowledge contained in the knowledge-based system** should be of primary concern when validating a knowledge base [Rus88].

Scalability happens to be another problem with current verification and validation approaches [PM96, PS94]. None of the current tools provide a framework for cutting down on the complexity of the knowledge-base before doing V&V. Due to knowledge bases being developed in a rapid prototyping environment with an iterative software development cycle, there is no practical and systematic software engineering methodology currently in place for their development. The situation is worsened due to the data-driven nature of expert systems, because as the number of rules of an expert system increase, the number of possible interactions across the rules increases exponentially. The complexity of each pattern in a rule compounds the problem of V&V even further. Defining any requirements or specifications up front in such a rapid prototyping and iterative development environment, even though they are desirable, becomes an impractical and moot question. Even if they were specified, as any software, conventional or knowledge-based, becomes more complex, common errors are bound to occur through misunderstandings of specifications and requirements. As a result, large expert systems tend to be incomprehensible, difficult to understand, and almost impossible to verify or validate.

It is our contention that in order to build reliable knowledge-based systems, it is important that the knowledge in the system be suitably abstracted, structured, and otherwise clustered in a manner which facilitates its understanding, verification, validation, maintenance, management and testing. It is therefore desirable to have an analysis tool that exposes a developer to the current semantics of the knowledge base in such a dynamically changing development environment, so that the knowledge base can be comprehended at various levels of detail. In this paper we would like to describe a prototype environment to address these issues based on our Multi-ViewPoint Clustering Analysis (MVP-CA) methodology.

The MVP-CA methodology addresses partitioning rule-based systems into a number of meaningful units prior to applying any V&V techniques.

Cluster analysis is a kind of unsupervised learning in which (a potentially large volume of) information is grouped into a (usually much smaller) set of clusters. If a simple description of the cluster is possible, then this description emphasizes critical features common to the cluster elements while suppressing irrelevant details. Thus, clustering has the potential to abstract from a large body of data, a set of underlying principles or concepts. This organizes that data into meaningful classes so that any verification, validation and testing can be performed meaningfully. Our approach hinges on generating clusters of rules in a large rule base, which are suggestive of mini-models related to the various sub domains being modeled by the expert system. These clusters can then form a basis for understanding the system both hierarchically (from detail to abstract) and orthogonally (from different perspectives). An assessment can be made of the depth of knowledge/reasoning being modeled by the system.

Overview of the MVP-CA Tool

Pragati's Multi-ViewPoint-Clustering Analysis (MVP-CA) tool provides such a framework for clustering large, homogeneous knowledge-based systems from multiple perspectives. It is a semi-automated tool allowing the user to focus attention on different aspects of the problem, thus providing a valuable aid for comprehension, maintenance, integration and evolution of knowledge-based systems. The generation of clusters to capture significant concepts in the domain seems more feasible in knowledge-based systems than in procedural software as the control aspects are abstracted away in the inference engine. It is our contention that the MVP-CA tool can form a valuable aid in exposing the conceptual software structures in such systems, so that verification, validation and testing efforts can be carried out meaningfully, instead of a brute-force or ad-hoc manner [BW88, CS88]. In addition, insight can be obtained for better reengineering of the software, to achieve run-time efficiency as well as reduce long-term maintenance costs. It is our intention to provide a comprehension aid base first, through our MVP-CA tool, for supporting all these software engineering activities. The MVP-CA tool consists of a Cluster Generation and a Cluster Analysis Phase. Together they help analyze the clusters so that these clusters can form the basis for any V&V, testing or maintenance activities. The multi-viewpoint approach utilizes clustering analysis techniques to group rules that share significant common properties and then it helps identify the concepts that underlie these groups. In the Cluster Generation Phase the focus is on generating meaningful clusters through clustering analysis techniques augmented with semantics-based measures. In this phase, the existing rule base, together with a concept focus list (in the form of a pattern elimination list) feeds into a front end interpreter. The interpreter parses the rule base and transforms it into an internal form required by the clustering tool. The clustering algorithm starts with each rule as a cluster. At each step of the algorithm, two clusters which are the most "similar" are merged together to form a new cluster. This pattern of mergings forms a hierarchy of clusters from the single-member rule clusters to a cluster containing all the rules. "Similarity" of rules is defined by a set of heuristic distance metrics for defining the distance between rules.

One of the most significant ways a user can effect the clustering process is through his choice of a distance metric. Distance Metric measures the relatedness of two rules in a rule base by capturing different types of information for different classes of expert systems [Cha86, MW95, Cla85]. There are five different distance metrics that we have implemented so far. Classification systems yield easily to a data-flow grouping and hence information is captured from the consequent of one rule to antecedent of other rules. This defines our ***data-flow*** metric. In a monitoring system since the bulk of domain information required for grouping is present in the antecedents of rules, the ***antecedent*** distance metric captures information only from the antecedents of rules. Alternatively, grouping the rule base on information from the consequents alone, gives rise to the ***consequent*** metric. The ***total*** metric is general enough and captures information from both sides of rules to take care of systems where a combination of the above programming methodologies exists. The ***ant-cons*** metric is a slight variation of the total metric in that it tracks information from the antecedents and consequents separately.

Capabilities of MVP-CA Technology

The prototype version of the MVP-CA tool is able to (semi) automatically expose the

- natural software architecture of the knowledge base
- verification & validation problems in the knowledge base, such as,
  - inconsistent rules,
  - redundant rules, and
  - incomplete coverage
- potentially restructurable software regions in the knowledge base and
- reducible testing regions in the knowledge base for generation of reduced test case suites

Natural Software Architecture of Knowledge Bases

The MVP-CA tool allows us to identify important attributes about the software architecture of a system by exposing important characteristics of selected clusters. In

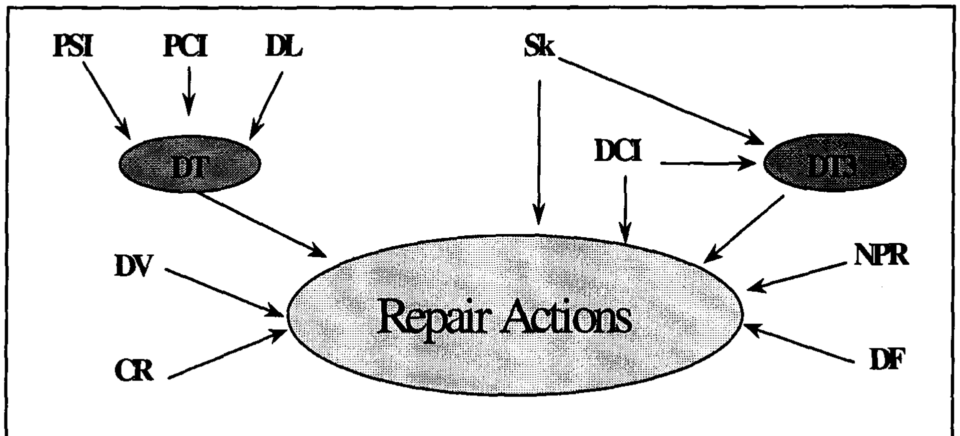

Figure 1: PAMEX Software Architecture

order to aid the selection of the cluster, the user is provided with information about some properties of the cluster, such as, the dominant pattern in the cluster, the parent link for the cluster, cohesiveness of the cluster, and one of the most important property, the stable patterns in the cluster. A stable pattern is very similar to a local variable in a procedure; that is, the pattern is not a member of any other cluster in the rule base. The MVP-CA tool presents the patterns in the rule base with statistics on its stable cluster, that is, the size of the cluster, frequency of the pattern in the rule base, etc. All this information coupled with a user's background knowledge of the domain, aids him/her in the selection of the important attributes or concept patterns in the knowledge base. By selecting the appropriate stable pattern(s) in the rule base, the user is able to study the different parameter relationships for that pattern in the cluster and its inter-dependencies among the different attributes of the system.

In Figures 1 and 2, we present the software architecture of the PAMEX and ESAA knowledge bases, two rule bases built by DOT [Meh94, Meh95a, Meh96]. *PAMEX* is a *pavement maintenance expert system* consisting of 327 rules written using the EXSYS expert system shell [Fed94b]. It is a diagnostic/monitoring system where symptoms are examined, causes are inferred for these symptoms, and remedies are advocated. Many components of pavement maintenance management are poorly structured and complex and do not yield to conventional programming approaches. The search space for possible causes of system deterioration is large enough to be unmanageable for verification, validation, and maintenance. However, PAMEX has been built with a considerable amount of thinking applied

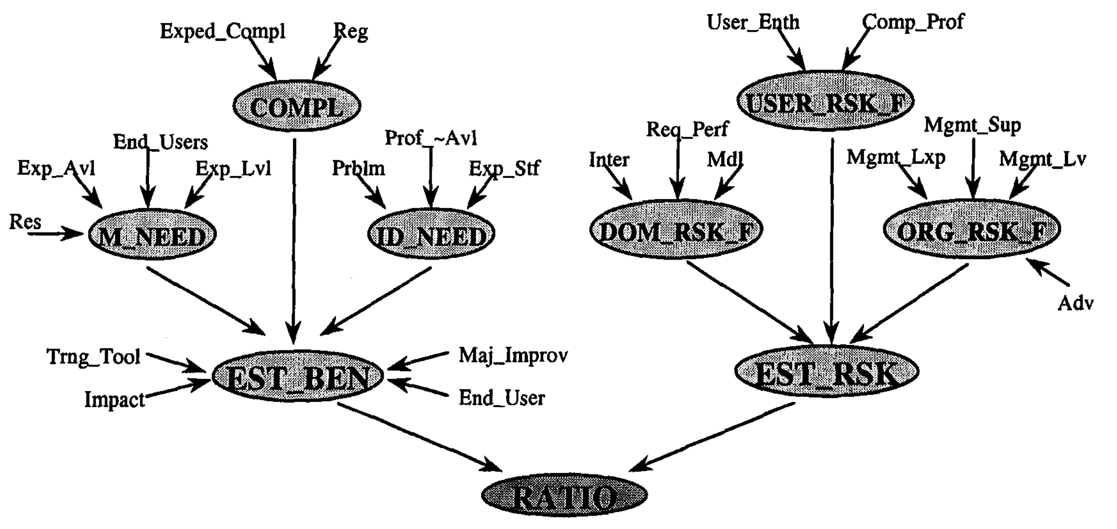

Figure 2: ESAA Software Architecture

Rule No.	*Description*
24	C_AN_T<>0 => AN_T_SAV(C_AN_T, ES_AN_T)
27	C_AN_T<>0 => RAW_AN_EXEC(AN_T_SAV, EXEC_T_SAV)

Stable Patterns: *C_AN_T, ES_AN_T, AN_T_SAV*

Figure 3: Conflict condition in ESAA

up front in the development process and is a very well-structured knowledge base. The second expert system from DOT, *Expert System Advocate's Advisor- ESAA*, consists of 68 EXSYS rules, 36 declared qualifiers (out of which only 27 were used in the rules), 27 variables, and 11 choices or conclusions. Even though ESAA is a small expert system, unlike PAMEX, this expert system did not exhibit careful up front software designing. Hence it provides a very good data point for us in terms of exposing the types of faults likely to be made when a rule base is developed in an ad hoc manner. We would like to point out that we were not a domain expert for any of these knowledge bases. In fact, for ESAA, the results were a revelation to the developers themselves because the rule base had been developed in a very ad hoc manner. Results from the two rule bases were a testimonial to the fact that such a software aid is a very necessary requirement considering the iterative style of development for typical expert systems. In fact, more recent work on telemetry knowledge-based systems, [Meh99], provides more direct evidence of the importance of the different types of information revealed by the MVP-CA tool. In this paper, however, we will restrict ourselves to expert systems used in the Department of Transportation.

Verification and Validation Capabilities

Rule clustering helps cut down the complexity of the knowledge base before searching for anomalous conditions. The MVP-CA tool has the capability to automatically flag clusters with *inconsistent rule pairs* – that is, when a rule pair has the same antecedent conditions but different conclusions. This is an anomaly in a forward chaining system, because the rule base asserts two different conclusions for the same premise. Rule base development environments generally do not flag such conflicts statically, leaving it to be resolved at run-time. This leads to non-predictable behavior of the rule base, as the rule firing for such a case would depend on the conflict-resolution strategy of the inference engine. Even if the system behaves as desired in a particular environment, these anomalies can lead to difficulty in porting the system to different expert system shell platforms [Lan90].

There are a number of anomalous conditions that surface quite early in the clustering process with the antecedent metric in ESAA. Examining group 69 in Figure 3 closely, it can be seen that two rules have the same premise --- C_AN_T <> 0 --- but different conclusions. This is an anomalous condition in the rule base as one of the rules (probably Rule 27) will never get a chance to fire (if the expert system shell uses rule ordering as its conflict resolution strategy). To correct the problem, one of these rules needs to be made more specific.

Also, a *redundant rule pair* condition is flagged from the MVP-CA tool quite easily, through the generation of rule clusters. Whenever two rules share the same conclusions and the conditions in the antecedent of one rule is a subset of the other rule's antecedent conditions; we mark those rules in the cluster as having a redundant condition. A redundant condition occurs in Figure 4 as ID_NEED is overspecified. Rule 66 is asserting that both PROF_~AVL and EXP_STF have to be affirmative in order to set the value of ID_NEED to "yes." However, Rule 67 is setting ID_NEED with only a subset of these conditions; in particular, it does not care for the value of EXP_STF. This conflict needs to be resolved with the help of the domain experts to bring the knowledge base to a consistent state. This anomaly was also discovered very quickly through the antecedent metric. In other words, one of the rules is more general.

Rule No.	*Description*
1	ID_NEED=Y => EST_BEN = 10
63	ID_NEED=N => EST_BEN=0
66	ID_NEED=U, PRBLM=CMPLX, PROF_AVL=Y, EXP_STF=Y => ID_NEED = Y
67	ID_NEED=U, PRBLM=CMPLX, PROF_AVL=Y => ID_NEED=Y
68	ID_NEED=U, PRBLM=CMPLX, EXP_STF=Y => ID_NEED=Y

Stable Patterns: *ID_NEED, PRBLM, PROF_AVL, EXP_STF*

Figure 4: Redundant Condition in ESAA

```
Rule No.        Rule No.        Description
   269  DCI.Cmb;CR.GE.5=>DT3=301
   287  DT-32;DT3-301;DV1.GE.20;DV13.L.40;DV15.L.25;DV17.L.30;
   270  DCI.SF=>DT3=301
   276  DT-31;DT3-301;DV1.GE.25;DV13.L.40;DV15.L.25;DV17.L.30;
   298  DT-33;DT3-301;DV1.GE.25;DV13.L.40;DV15.L.20;DV17.L.30;
   289  DT-32;DT3-301;DV1.L.20;DV13.L.40;DV15.L.25;DV17.L.30;
   275  DT-31;DT3-301;DV1.GE.25;DV13.L.40;DV15.GE.25;DV17.L.30;
   278  DT-31;DT3-301;DV1.L.25;DV13.L.40;DV15.L.25;DV17.L.30;
   300  DT-33;DT3-301;DV1.L.25;DV13.L.40;DV15.L.20;DV17.L.30;
   297  DT-33;DT3-301;DV1.GE.25;DV13.L.40;DV15.GE.20;DV17.L.30;
   277  DT-31;DT3-301;DV1.L.25;DV13.L.40;DV15.GE.25;DV17.L.30;
   286  DT-32;DT3-301;DV1.GE.20;DV13.L.40;DV15.GE.25;DV17.L.30;
   299  DT-33;DT3-301;DV1.L.25;DV13.L.40;DV15.GE.20;DV17.L.30;
   288  DT-32;DT3-301;DV1.L.20;DV13.L.40;DV15.GE.25;DV17.L.30;

   Number of Stable Patterns = 1
      Pattern#   Pattern Description
      383           301

The following expressions are common:
DV13 < 40
DV17 < 30
DT3  = 301
```

Figure 5: Restructurable Region in PAMEX

Unless the inference engine is geared towards firing the more specific rule first, the more general rule will always fire, masking the existence of the more specific rule altogether. Thus, the firing of these rules is going to depend on the conflict-resolution strategy of the inference engine that would tend to make the code difficult to port. Such conflicts need to be resolved with the help of the domain experts to bring the knowledge base to a consistent state.

The multi-view point clustering of a rule base is capable of grouping together rules that are similar in content and form. This can be suggestive of *incomplete areas of reasoning* in the knowledge base because rules with similar premises or similar conclusions come together into a single group. Semantic incompleteness is easier to detect if rules addressing similar sub domain information can be brought together into a group. If all values for an attribute have been specified as input to our tool, then the current version of the tool is equipped to perform a completeness check to ascertain that all declared values for the attributes have been addressed.

Even though the detection of such anomalous conditions could have been performed in a brute force manner, the MVP-CA tool provides a scalable solution to the detection of these problems by cutting down the complexity of a rule base and providing rule clusters as the primary units of analysis. Moreover, an added benefit of seeing these anomalies in a clustered environment is that the user can study the context for these rules before making any corrections.

Restructurable Regions in Knowledge Bases

In the rapid development environment of rule bases, the "add-a-rule-each-time" philosophy of keeping the rule base current with respect to new incoming requirements, usually results in the decision tree becoming very lop-sided with time. In other words, there may be conditions in several rules that are being tested at a lower level in the tree, which may benefit from being factored out and pulled to a higher level of the decision tree. Such conditions need to be identified and the rules restructured for future efficiency.

In our experiments with PAMEX, we found that even though the rule base had been built with very good software engineering principles, there were regions that could use knowledge on common factoring of expressions that were exposed through the MVP-CA tool. When a user inquires about restructuring possibilities for a pattern or pattern combination, such as DT3=301 in Figure 5, the MVP-CA tool is able to automatically identify common expressions such as, DV13 < 40 and DV17 < 30, across the rules for this stable group, so that these expressions can be factored out from the rules and tested at a higher level or earlier in the tree. Such a restructuring would have several benefits. First, the lower level of the tree would be

Rule No.	*Antecedent*
27	*DT = 11; Sk=L; DV12 >= 15; DV2 < 15;=>...*
28	*DT = 11; Sk=L; DV12 < 15; DV2 < 15;=>..*
31	*DT = 12; Sk=L; DV12 >= 15; DV2 < 15;=>..*
32	*DT = 12; Sk=L; DV12 < 15; DV2 < 15;=>..*

Stable Patterns: *DV12*

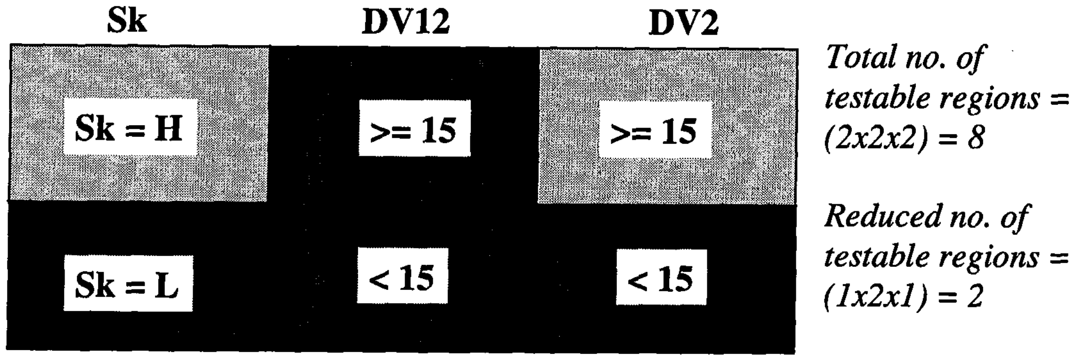
Sk
DV12
DV2
Sk = H
>= 15
>= 15
Sk = L
< 15
< 15
Total no. of testable regions = (2x2x2) = 8
Reduced no. of testable regions = (1x2x1) = 2

Figure 6: Equivalence Partitioning for PAMEX with respect to DV12

simplified leading to a more understandable rule base. Second, this would lead to an increase in the runtime efficiency of the rule base by reducing the number of times these tests have to be performed.

Reducible Testing Regions in Knowledge Bases

Current practice for judging the operational ability of a rule base is by subjecting the rule base to a set of representative test case suites to test as many decision paths as possible. Exhaustive testing is not feasible due to the combinatorial explosion involved in testing all possible paths for a rule base. Through the clustering analysis MVP-CA offers a smart solution for designing test cases. By providing suitable variables as focal points for formation of sub knowledge bases to be tested, the MVP-CA tool reduces the computational complexity of test generation [Fed97a, Fed94b].

Automatic detection routines in the MVP-CA tool are able to identify these equivalence regions for partitioning. Thus, instead of testing for all values in these regions, one can test for only one representative value from the region (or use the whole region symbolically), as shown in Figure 6. By factoring out mutually exclusive regions based on special purpose stable variables, proliferation of test regions could be considerably controlled. Identification of such variables is eased substantially through the multi-viewpoint clustering analysis techniques.

Related Work

Extraction of meta-knowledge for the purposes of comprehending and maintaining expert systems has been an accepted norm for some of the best-known fielded systems. Systems such as XCON [BO89, SBJ87] had to develop a new rule-based language, RIME, and rewrite XCON-in-RIME to facilitate its maintenance. XCON-in-RIME is supposed to make the domain knowledge more explicit, both in terms of restructuring the rules and in terms of exposing the control structure for firing of the rules. Meta-Dendral [Buc79] tries to resolve the bottleneck of knowledge acquisition through automatic generation of rule sets so as to aid the process of formation of newer scientific theories in mass spectroscopy. TEIRESIAS [BS85] uses meta-rules to encode rule-based strategies that govern the usage of other rules. For this purpose it generates a set of rule models that are then used to guide this effort by being suggestive of both the content and form of the rules. These systems, even though they are very domain-specific, formed the inspirational basis for our work.

Jacob and Froscher [JF86, JF90] have attempted to cluster knowledge bases in order to abstract and structure the knowledge in them. Their rule grouping is primarily based on identifying the group of rules that use or produce the same set of facts. Hence, their clustering strategy presumes and gives preference to a data-flow dependency among rules. During our experience with different types of knowledge base applications we have been able to develop a richer repertoire of different types of rule interdependencies in the MVP-CA tool. Lindell and others [LIN87, LVR87] have also tried to cluster knowledge-bases based on keywords, but their approach is limited to cases where there are appropriate keywords in the knowledge-base to cluster around. There has been research in parallelizing user-specified rule groups with a view to improving the real-time characteristics of an expert system architecture by Chen and Poole [CP94]. However, the issues focus on the runtime characteristics of the system, such as optimal scheduling and minimizing the overhead associated in the activation frameworks for rule groups. The issues we face in static analysis are quite different.

There is as yet no automated clustering approach that tries to structure large, monolithic knowledge bases for the purpose of verifying, validating and reusing them. Most verification and validation tools for knowledge-based systems, such as ONCOCIN [SSS82], KB-Reducer [Gin88], Lockheed's CHECK [NPLP87], CASNET [WKAS78], have been research projects which do static analysis of the knowledge base based upon either a table-driven or a graph-based approach. Preece's COVER [Pre92] also carries out a number of verification checks. In COVER, rules have to be in the first-order logic form before they can be input. Checks are carried out on a dependency graph of the rule base. Since the techniques for verifying and validating rule bases in such systems rely on obtaining the whole rule base as input and applying the V&V techniques on the entire rule-base, they become impractical for large knowledge bases. They do not offer a scalable solution for V&V of large knowledge-based systems. MVP-CA advocates breaking down a knowledge-base **first,** before applying V&V techniques. Once the rule base has been clustered, the V&V techniques can be applied much more effectively on the decomposed system. In addition, the clustering algorithm used by the MVP-CA tool incorporates both statistical and semantics-based measures to cluster - a feature that is not present in standard clustering algorithms [JP73, Jar78, GK78a, GK78b]. This provides an added advantage for viewing the V&V results in the semantic context. For example, when two redundant or inconsistent rules are flagged, the results are seen along with the other associated rules in the cluster. Once the semantic underpinnings of such clusters are identified, more meaningful and effective corrections can be applied to the knowledge base. Also, exposure of the software architecture of the knowledge base as well as aiding in the testing efforts of the knowledge base could not have been possible without decomposing the knowledge base first. Due to the declarative style of programming in knowledge-based systems, generation of clusters to capture significant concepts in the domain of knowledge-based systems seems more feasible than it would be for procedural software. MVP-CA tool exploits this aspect of knowledge bases so as to software engineer them effectively.

Conclusions

Our tool demonstrates the feasibility of generating different clusterings to expose different legitimate perspectives of a knowledge-based system. Studying parameter relationships and inter-dependencies uncovered through our clustering techniques can make better software engineering decisions regarding the design of the knowledge base. In particular, the knowledge-base can be restructured for better runtime efficiency as well as better comprehensibility and long-term maintainability. In addition, clustering the knowledge base from multiple viewpoints provides a structured and more efficient approach to test case generation. A structured approach to testing, management and maintenance of knowledge-based systems would go a long way towards dispelling the myth that expert systems are inherently unreliable and that nothing can be done about it. An integrated environment for expert system verification and validation, such as is proposed by MVP-CA, would overcome this barrier, opening expert systems for a broad range of important applications.

References

[BO89] V. E. Barker and D. E. O'Connor. Expert Systems for Configuration at Digital: XCON and beyond. Communications of the ACM, 32(3), March 1989.

[BW88] K. L. Bellman and D. O. Walter. Analyzing and correcting knowledge-based systems requires explicit models. In AAAI-88 Workshop on Verification, Validation and Testing of Knowledge-Based Systems, July 1988.

[BS85] B.G. Buchanan and E.H. Shortliffe. Knowledge Engineering. In Rule-Based Expert Systems, chapter 7, pages 149-158. Addison Wesley Publishing Co., 1985.

[Buc79] B.G. Buchanan. Issues of Repesentation in Conveying the Scope and Limitations of Intelligent Assistant Programs. In J.E. Hayes, D. Michie, and L.I. Mikulich, editors, Machine Intelligence, pages 407-425. John Wiley \& Sons, 1979.

[Cha86] B. Chandrasekharan. Generic tasks in knowledge-based reasoning: High-level building blocks for expert systems design. IEEE Expert, Fall 1986.

[CP94] I. Chen and B. L. Poole. Performance Evaluation of Rule-Grouping on a Real-Time Exoert System Architecture. IEEE Transactions on Knowledge and Data Engineering: Vol.6. No. 6. Pages 883-891. December 1994.

[Cla85] W. J. Clancey. Classification problem solving. In Proceedings, National Conference on Artificial Intelligence, pages 49-55, 1985.

[CS88] C. Culbert and R. T. Savely. Expert System Verification and Validation. In Proceedings, Validation and Testing Knowledge-Based Systems Workshop, August 1988.

[Fed97a] Federal Highway Administration. Expert System Verification, Validation and Evaluation Handbook: Version 2 Report No: FHWA-RD-95-196, Jan 1997.

[Fed94b] Federal Highway Administration. Highway Applications of Expert Systems: Recent Developments and Future Directions, 1994.

[Gin88] A. Ginsberg. A new approach to checking knowledge bases for inconsistency and redundancy. In Proceedings of the Seventh Annual National Conference on Artificial Intelligence, pages 585-589, 1988.

[GK78a] K. C. Gowda and G. Krishna. Agglomerative clustering using the concept of mutual nearest neighborhood. Pattern Recognition, 10(2):105-112, 1978.

[GK78b] K. C. Gowda and G. Krishna. Disaggregative clustering using the concept of mutual nearest neighborhood. IEEE Transactions on Systems, Man, and Cybernetics, SMC-8(12):888-895, December 1978.

[Jar78] R. A. Jarvis. Shared near neighbor maximal spanning trees for cluster analysis. In Proceedings of the 4th Internatinal Joint Conference on Pattren Recognition, pages 308-313, November 1978.

[JF86] R. J. K. Jacob and J. N. Froscher. Developing a Software Engineering Methodology for Knowledge-Based Systems. Technical Report 9019, Naval Research Laboratory, Washington, D.C., December 1986.

[JF90] R. J. K. Jacob and J. N. Froscher. A software engineering methodology for rule-based systems. IEEE Transactions on Knowledge and Data Engineering, 2(2):173-189, June 1990.

[JP73] R. A. Jarvis and E. A. Patrick. Clustering using a similarity measure based on shared near neighbors. IEEE Transactions on Computers, C-22(11):1025-1034, November 1973.

[Lan90] C. Landauer. Correctness principles for rule-based expert systems. Journal of Expert Systems with Applications, 1:291-316, 1990.

[Lin87] S. Lindell. Keyword Cluster Algorithm for Expert System Rule Bases. Technical Report SD-TR-87-36, The Aerospace Corporation, El Segundo, CA., June 1987.

[LVR87] K. Lindenmayer, S. Vick, and D. Rosenthal. Maintaining an Expert System for the Hubble Space Telescope Ground Support. In Proceedings, Goddard Conference on Space Applications of Artificial Intelligence and Robotics, pages 1-13, May 1987.

[Meh94] M. Mehrotra. Application of multi-viewpoint clustering analysis to a highway maintenance system. In Notes for the AAAI-94 Workshop on Verification, Validation and Testing of Knowledge-Based Systems, August 1994.

[Meh95a] M. Mehrotra. Requirements and capabilities of the multi-viewpoint clustering analysis methodology. In Notes for the IJCAI-95 Workshop on Verification, Validation and Testing of Knowledge-Based Systems, Montreal, Canada, August 1995.

[Meh96] M. Mehrotra. Application of multi-viewpoint clustering analysis to an Expert Systems Advocate Advisor. Technical Report FHWA-RD-97022, Federal Highway Administration, Pragati Final Report, Yorktown, VA., August 1996.

[Meh99] M. Mehrotra, S. Alvarado and R. Wainwright. Laying a Foundation for Software Engineering of Knowledge Bases in Spacecraft Ground Systems. To Appear in Proceedings of FLAIRS-99 Conference to be held May 3-5th, 1999 in Florida.

[MW95] M. Mehrotra and C. Wild. Analyzing knowledge-based systems using multi-viewpoint clustering analysis. Journal of Systems and Software, 29:235-249, 1995.

[NPLP87] T.A. Nguyen, W.A. Perkins, T.J. Laffey, and D. Pecora. Knowledge-base verification. AI Magazine, 8(2): 69-75, Summer 1987.

[PM96] R.T. Plant and S. Murrell. A survey of tools for validation and verification 1987-1996. Decision Support Systems, to appear, 1996.

[PS94] R.T. Plant and J.P. Salinas. Expert systems shell benchmarks: The missing comparison factor. Information and Management, 27:89-101, 1994.

[Pre92] A. D. Preece. Principles and Practice in Verifying Rule-Based Systems. The Knowledge Engineering Review. Vol. 7(2). Pages 115-141.

[Rus88] J. Rushby. Quality Measures and Assurance for AI Software. Technical Report NASA CR-4187, SRI International, Menlo Park, CA., October 1988.

[SBJ87] E. Soloway, J. Bachant, and K. Jensen. Assessing the Maintainability of XCON-in-RIME: Coping with the Problems of a VERY Large Rulebase. In Proceedings of AAAI-87, July 1987.

[SSS82] M.Suwa, S.C. Scott, and E.H. Shortliffe. An approach to verifying completeness and consistency in rule-based expert system. AI Magazine, 3(4):16-21, Fall 1982.

[WKAS78] S.M.Weiss, C.A.Kulikowski, S. Amarel, and A. Safir. A model-based method for computer-aided medical decision-making. Artificial Intelligence, 11:145-172, 1978.