KASER: Knowledge Amplification by Structured Expert Randomization

Stuart H. Rubin, *Senior Member, IEEE*, S. N. Jayaram Murthy, *Member, IEEE*, Michael H. Smith, *Senior Member, IEEE*, and Ljiljana Trajković, *Senior Member, IEEE*

***Abstract*—In this paper and attached video, we present a third-generation expert system named Knowledge Amplification by Structured Expert Randomization (KASER) for which a patent has been filed by the U.S. Navy's SPAWAR Systems Center, San Diego, CA (SSC SD). KASER is a creative expert system. It is capable of deductive, inductive, and mixed derivations. Its qualitative creativity is realized by using a tree-search mechanism. The system achieves creative reasoning by using a declarative representation of knowledge consisting of object trees and inheritance. KASER computes with words and phrases. It possesses a capability for metaphor-based explanations. This capability is useful in explaining its creative suggestions and serves to augment the capabilities provided by the explanation subsystems of conventional expert systems.**

KASER also exhibits an accelerated capability to learn. However, this capability depends on the particulars of the selected application domain. For example, application domains such as the game of chess exhibit a high degree of geometric symmetry. Conversely, application domains such as the game of craps played with two dice exhibit no predictable pattern, unless the dice are loaded. More generally, we say that domains whose informative content can be compressed to a significant degree without loss (or with relatively little loss) are symmetric. Incompressible domains are said to be asymmetric or random. The measure of symmetry plus the measure of randomness must always sum to unity.

***Index Terms*—Expert systems, knowledge amplification, machine learning, object-oriented design, randomization.**

I. Introduction to Randomization

THE THEORY of randomization was published first by Chaitin [1] in 1975. That work may be seen as a consequence of Gödel's Incompleteness Theorem [2] because it shows that were it not for essential incompleteness, then a universal knowledge base could be constructed—one that need not employ other search than that used for information retrieval. However, Lin and Vitter [3] also proved that learning must be domain-specific in order to be tractable. This fundamental need for domain-specific knowledge is in keeping with the "unsolvability of the randomization problem" [4]. This proof introduced the concept of knowledge amplification. Production rules are expressed in "situation → action" form. Such rules, once discovered to be in error, are corrected through acquisition. Conventionally, a new rule must be acquired for each correction. This process may be called "linear learning." While knowledge may be defined to be synonymous with a corpus of rules in the realm of information technology, this paper introduces a more precise meaning for the term—one that is consistent with first principles of randomization [1], [4]. Here, we define knowledge to be a relative term, which refers to any semantic randomization of information (i.e., data).

The key to breakthroughs in the creation of intelligent software is in dealing with the knowledge acquisition bottleneck [5]–[7]. Learning how to learn depends fundamentally on representing knowledge in the form of a distributed society of experts. That is, domain-specific representations must be networked to achieve greater generality. Minsky's work [8] on brain-like networked representations of knowledge led to the development of intelligent agent architectures. Furthermore, it has been independently shown [4], [9], [10] that in order for machine-learning paradigms to be scalable, the representational formalism itself must be included in the definition of domain-specific learning. Relatively recent evidence from neuroscience shows that there are parts of the brain that control other parts with far less restrictions on the kind of procedures that can be called "brain like" than is allowed by current connectionist theory [11].

KASER is a knowledge amplifier based on the principle of structured expert randomization. This principle refers to the use of basic (fundamental) knowledge in the capture and reduction of a larger, dependent space of knowledge (not excluding self-reference). In KASER, the user supplies declarative knowledge in the form of a semantic tree using single inheritance.

Conventional expert systems generate cost curves below the breakeven line. In conventional expert systems, cost increases with scale and the increase is never better than linear. In the case of KASER, the cost decreases with scale and is always better than linear, unless the domain is asymmetric (random). Perfectly (asymmetric) random domains are trivial constructs and are not encountered in the construction of practical applications [1], [12]. Conversely, perfectly symmetric (nonrandom) domains are also trivial and are also not found in practice [1].

Manuscript received June 2, 2002; revised November 7, 2003. This work was supported in part by the U.S. Air Force by adopting KASER for the generation of diagnostics for the AF F-16 AN/APG-68 radar system. This paper was recommended by Associate Editor G. Biswas.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors. This includes a video named KASER.gif and a README.txt file that contains instructions on how to best view the video. KASER.gif is a 1.9 Mbyte video file in GIF format.

S. H. Rubin is with the SPAWAR Systems Center, San Diego, CA 92152-5001 USA (e-mail: stuart.rubin@navy.mil).

S. N. J. Murthy is with the Department of Computer Science, Central Michigan University, Mt. Pleasant, MI 48859 USA (e-mail: S.N.Jayaram.Murthy@cmich.edu).

M. H. Smith is with the BISC Program, University of California, Berkeley, CA, 94720 USA (e-mail: m.h.smith@ieee.org).

L. Trajkovic is with the School of Engineering Science, Simon Fraser University, Burnaby, BC V5A 1S6 Canada (e-mail: ljilja@cs.sfu.ca).

Digital Object Identifier 10.1109/TSMCB.2004.835081

In other words, a perfectly random domain would not have embedded patterns (true random numbers), while a perfectly symmetric domain would be infinitely compressible (free of information content). Clearly, such constructs are strictly artificial. Again, we find that the more symmetric the operational domain, the less the cost of knowledge acquisition. It always holds that the virtual rule space is much larger than the real rule space.

The KASER is patent-pending. In 2002, the U.S. Navy's SPAWAR Systems Center (SSC), San Diego filed a patent application (NC 83 014) with the USPTO, which details further claims pertaining to the KASER system [30].

This paper is organized as follows: In Section II, we introduce the notion of qualitative fuzziness. Details of KASER's heuristic search algorithm are described in Section III. Declarative objects are discussed in Section IV. Next, an example of KASER is presented in Section V. Finally, conclusions are given in Section VI.

II. Qualitative Fuzziness

KASER defines a production system that can automatically acquire a virtual rule space that is exponentially larger than the actual rule space. If one assumes that a correctly induced rule lies within the indefinite space of a valid rule, then one can represent the inductive process using a probability density function. Here, the random variable X has the error density function

$$f(x) = ce^{-a|x|}, \quad -\infty < x < \infty$$

where $a > 0$, and c is a suitable constant. Since $f(x)$ is a density function, it follows that

$$\int_{-\infty}^{\infty} f(x)dx = 1$$

so that

$$\begin{aligned} c\int_{-\infty}^{\infty} e^{-a|x|}dx =& c\left[\int_{-\infty}^{0} e^{-a(-x)}dx + \int_{0}^{\infty} e^{-a(x)}dx\right] \\ =& c\frac{e^{ax}}{a}|_{-\infty}^{0} + c\frac{e^{-ax}}{-a}|_{0}^{\infty} = \frac{2c}{a} = 1. \end{aligned}$$

Then, $c = (a/2)$ with a likelihood of error (i.e., the chance that any one induction is erroneous) defined by

$$\frac{e^{-x}}{2}, \quad x \geq 0.$$

Moreover, the generalization mechanism will have bounded error. Other than for a straightforward user-query process, it will operate without any *a priori* knowledge supplied by the user.

To begin, we define a production rule to be an ordered pair whose first member is a set of antecedent predicates and whose second member is an ordered list of consequent predicates. Predicates can be numbers or words [13], [14]. Antecedents are represented by a conjunct of Boolean functions (or equivalent); while, consequents are represented by a sequence of procedural actions (or equivalent).

Previously unknown words or phrases can be recursively defined in terms of known ones. For example, the moves for the Queen in chess (unknown) can be defined in terms of the union of the moves for a Bishop and for a Rook. This is a union of property lists. Other set operations may likewise be used (intersection, difference). The use of fuzzy set operators here ("almost the same as") pertains again to computing with words [13]–[15].

KASER systems can be classified as Type I and Type II, depending on their characteristics. In a Type I KASER, words and phrases are entered through the pull-down menus. The user is not allowed to enter new words or phrases if an equivalent semantics already exists in the menu. In that manner, semantically identical concepts ("Hello" and "Hi") are not ascribed a distinct syntax, which would otherwise serve to dilute the efficiency of the learning mechanism. In a Type II KASER, distinct syntax may be equated to yield the equivalent normalized semantics. The idea in a Type II KASER is to ameliorate the inconvenience of using a data entry menu with scale. For example, a child may ask: "What is a bird?" The reply is: "It is an animal that flies." His subsequent question is: "What is an animal?" with the reply: "It is a living thing." It is understood that the reply, "It is a living thing." is atomic. It defines the semantics for the preceding three phrases. The associated semantics may be subject to evolution. In a Type II KASER, selection lists are replaced with semantic equations from which the list problem is automatically solved. In this paper we restrict our presentation to a Type I KASER.

III. Heuristic Search Algorithm for Knowledge Acquisition

KASERs can amplify a knowledge base. In this section, we present a relatively high-level view of the KASER heuristic search algorithm. It represents an advance in the design of intelligent systems because of its capability for symbolic learning and qualitative fuzziness. We have included a supplementary file named KASER.gif. KASER.gif is a 1.9 Mbyte video file in GIF format that illustrates the operation of KASER using two examples of knowledge bases. Also attached is a README.txt file that contains instructions on how to best view the video. This will be available at http://ieeexplore.ieee.org. The reader is encouraged to view the video Attachment 1, which clarifies the knowledge acquisition algorithm and illustrates the KASER system.

One of the principle advantages afforded by conventional expert systems is that they provide for a clean separation of the inference engine from the knowledge base, which facilitates maintenance. Here, the context (i.e., current instance of the state space) may match the candidate rule antecedent (i.e., the conditional conjunct in the schema, *If condition then action sequence*, or consequent), in which case an agenda mechanism is used to decide which matched rule to execute, or *fire* (most-specific match, first to match, chance match.) [16], [17]. KASERs follow the same rule-firing principle—only the pattern-matching algorithm is necessarily more complex and embeds the conventional approach as its degenerate case. For example, conventional expert systems cannot match a context (the object predicate "CAR") with an isomorphic rule antecedent (the object predicate "AUTOMOBILE"). Nor can they match this context against an antecedent instance (the object predicate "HONDA") or against an antecedent generalization (the object predicate, "VEHICLE"). KASER's make use of a dynamic tree structure,

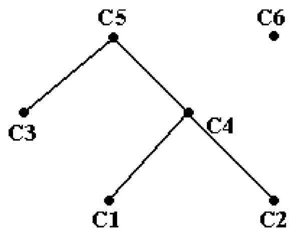


Fig. 1. Antecedent predicate grammar.

which defines the relation among conceptual predicates (e.g., FORD and HONDA are semantically related in that both are instances of AUTOMOBILE) and makes such a computing with words possible [4], [13]–[15].

In a KASER, the search for any contextual match must be ordered along gradients of nonincreasing validity of match. This is necessary to insure the nondecreasing performance of the learning mechanism. First, notice that the greater the number of edges separating two conceptual predicates in a tree (Fig. 1), the weaker the relation. (The absence of a path defines an infinite separation.) Predicate trees are not directed because the traversal algorithm needs to be capable of moving from sibling to sibling. Next, observe that instantiation preserves validity, whereas generalization allows for creativity. For example, if a given engine (e.g., a ramjet) can run on any "FUEL," then it will run on any instance of "FUEL" (ALCOHOL). Conversely, if an automobile engine can run on "GASOLINE," then it may or may not run on some generalization of "GASOLINE" (a sibling node, which by definition is an instance of a generalization) ("METHANE"). Of course, multiple predicates will need to be matched at most trials. Some matches will be direct as is the case with conventional expert systems, others will be instantiations to a greater or lesser degree, and some will necessarily be generalizations to a greater or lesser degree. Clearly, generalizations are matched as a last resort because they will greatly lower the probability of a valid result. The approach taken here fails softly, which also serves to differentiate it from conventional expert systems.

To begin with, the predicate-matching algorithm now has predicate generalizations and specializations included in its search space. A context is said to *cover* a rule antecedent just in case every predicate in that antecedent is matched by a predicate, an instantiated predicate, or a generalized predicate in the supplied context. Whenever the most specific rule is so matched and is the most recent one to be so matched given a tie, then that rules consequent is executed, or fired. The essential task of the algorithm is to minimize the number and degree of erroneous matches in any contextual covering. This is accomplished by taking any predicate instance for a match before any predicate generalization. Given a finite search depth, such a partial ordering is always possible. Moreover, the KASER interacts with the end user to incrementally acquire detailed knowledge in the form of predicate trees and of course rules that take their definition thereof. As the antecedent predicate tree grows in depth and breadth, the error in the aforementioned predicate matching algorithm tends toward zero. Thus, given any closed domain, the KASER will over time converge toward zero error in the creation of new knowledge.

We introduce the reader to the complex KASER algorithm using a very high-level, ten-step conceptual design. Each step will then be described in more detail in what follows.

1) A context is specified by the user and is subsequently, automatically reduced to the fewest terms.
2) The specific stochastic measure, which represents the number of levels of predicate instantiation needed to find a predicate match, is computed.
3) The matching process exits with success or failure, which is defined by temporal and spatial interrupts.
4) The general stochastic measure, which represents the number of levels of predicate generalization needed to find a predicate match, is computed.
5) An ordered least specific followed by a least general gradient search to match the constituent contextual predicates is performed.
6) The matching process exits with success or failure, based on the execution of the defined interrupts.
7) Where the system predicts an action sequence that is deemed to be erroneous, the knowledge engineer may specify an appropriate consequent, which in turn may additionally query the user to elicit further context.
8) An explanation subsystem is provided that uses specialization and generalization based similarity matches in the predicate trees to immediately provide the user with metaphorical explanations.
9) Given the inexact or fuzzy qualitative nature of the inductive process for rule generation, it follows that distinct distributed system outputs can, in principle, be fused to yield a network of even faster learning systems.
10) The user can manually direct the system to perform a more informed or deeper search than can sometimes be automatically achieved through the insertion of expansion markers, which inform the inference engine to pattern match all expansions of the designated predicate(s) to the designated depth.

We further clarify the complex algorithm for the firing mechanism by using an example. Note that tree traversal operations may be applied to the context and/or the antecedent predicates to make a more efficient algorithm. Also, error estimates are dynamically based on experience (domain-specific feedback) with each type of match.

1) First, antecedent menus are used to specify a contextual conjunct. For example, such a context might specify that my car is a Honda and that it will not start (HONDA & NOT_START). The antecedent menus allow navigation of a generalization tree to arrive at the desired conjunct (VEHICLE → AUTOMOBILE → HONDA). The context is then iteratively normalized (reduced to the fewest terms) using the tree grammar. Such grammars may be created semi-automatically [18]. The reduction rules, which are iteratively applied in any order—allowing for concurrent processing are

   a) Given that S is a generalized predicate in the antecedent tree and $AB, C\dots$ are instance predicates in this tree, where $S \rightarrow A \mid B \mid C\dots$, then replace $A, B, C\dots$ with S just in case all of the right-hand side (RHS) is present in the context. This step should be iteratively applied to conclusion before proceeding to the next step.

b) Given $S \rightarrow A\ldots$ and $A \rightarrow B\ldots$ and $B \rightarrow C\ldots$ then if S, A, B, C are all present in the context, then remove A, B, C because they are subsumed by S. It is never necessary to repeat the first step after conclusion of the second.

This results in contextual normalization being realized in linear time in terms of the number of conjuncts and the depth of search.

2) Secondly, compute the specific stochastic measure, which is indicative of the degree of predicate instantiation taken in the antecedent tree. For example, consider the antecedent grammar: $C5 \rightarrow C3 \mid C4$; $C4 \rightarrow C1 \mid C2$, where $C1$, $C2$, $C3$, $C4$, and $C5$ are labeled conjuncts (Fig. 1). Their level in the tree relates predicates. For example, at level 0, one might have the object predicate "AUTOMOBILE" (e.g., predicate $C5$ in Fig. 1), while level 1 would have the object predicate, "HONDA" (e.g., predicate $C3$ in Fig. 1), where Honda is an instance of automobile (and terminal in this case). In what follows, the first pair of braces $\{\}$ enclose the specified context, while the second pair of braces enclose a specified rule-base antecedent, which will be tested to see if it is covered (fired) by the specified context. Disjuncts are separated by commas and otherwise conjunction is implied.

a) $\{C3\ C1\}\{\{C3\},\{C2\ C3\}\}$ covers and matches the first $\{C3\}$ at level 0. That is, the rule's antecedent predicates need not be expanded for a match. The first covered match, if any, which does not have a covered superset, is the one to be fired—a result that follows from the method of transposition. Thus, the associated rule is fired.

b) $\{C3\ \ C1\}\{\{C5\},\{C2\ \ C3\}\}$ matches nothing at the level 0 expansion, so the rule antecedent is expanded with the result, $\{C3\ C1\}$ $\{\{*C5\,(C3,C4)\},\{*C2\,*C3\}\}$ where the $C2\,C3$ are both primitives and $*Ci$ can be matched, but not expanded again by definition (the $*$ designation is for the sake of computational efficiency) (Fig. 1). Here, $\{C5\}$ is matched at level 1 by $C3$ in the context (one level removed from the starting point in the search tree).

c) $\{C3\ \ C6\}\{\{C2\},\{C5\ \ C6\}\}$ matches nothing at the level 0 expansion, so the rule antecedent is expanded with the result, $\{C3\ C6\}$ $\{\{*C2\},\{*C5\ (C3,C4)\ *C6\}\}$, which matches at level 1 because ($C3$ OR $C4$) AND $C6$ were matched (Fig. 1). An attempt was not made to expand $C6$ because it was prematched by the existing context. This economy is possible as a result of prenormalizing the context.

d) The method of transposition is applied to the rule antecedent in the above step yielding the result, $\{\{C5\ C6\},\{C2\}\}$. Transposition insures that the most recent match will be taken in the event of a tie.

e) Each matched rule antecedent fires a rule consequent, which if not primitive, matches exactly one row header in the algorithm (a unique integer) and step (2) iterates.

f) Maintain a global sum of the number of levels of expansion for each row for each consequent term. The specific stochastic measure is the maximum of the number of levels of expansion used for each consequent term.

3) Next, exit the matching process with success (for a row) or failure based on reaching the primitive levels, a timer-interrupt, a forced interrupt, and/or using the maximum allocated memory.

4) Then, if a sequence of consequent actions has been attached, the sequence is pushed onto a stack in reverse order such that each item on the stack is expanded in a depth-first manner. A parenthesized sequence of actions will make clear the hierarchy. Set the general stochastic measure (GSM) to zero.

As an example defining the use of stochastic measure, observe that Toyota and Ford are members of the class, *car*. If Toyota is generalized to obtain car, which is subsequently instantiated to obtain Ford (an analog), then the general and specific stochastic measures would both be equal to one. The general stochastic measure (GSM) represents the number of levels of expansion for a term in one direction and the specific stochastic measure (SSM) represents the number of levels of expansion from this extreme in the opposite direction needed to get a match. The final general (specific) stochastic is taken as the maximum general (specific) stochastic over all terms. The GSM is a measure of generalization and again the SSM is a measure of instantiation.

5) If a match is not found, then, since an expanded rule antecedent is already present, the context is expanded in a breadth-first manner (if enabled by the level of permitted generalization). The search here is for a minimal generalization across all involved predicates to obtain a match. The general stochastic measure, a measure of validity where the aforementioned "tree inductions" are involved, is then computed. (The general stochastic measure is initialized to GSM, where the current context defines the initial presentation of the problem.)

a) While a specialized match was sought in step (2), a generalized match is sought here. Expanding the context can lead to redundancies. For example, $\{*C1\ *C2\ *C3\ *C4\ C1\ C2\}$. Here, the solution is simply not to include any term that is already in the (expanded) context. Stochastic accuracy is thus preserved. Any method that does not preserve stochastic accuracy is not used.

b) $\{C5\}\{\{*C3\},\{*C2\ *C3\}\}$ failed to be matched in step (2). Hence, a level 1 expansion of the context is taken:

$\{*C5\ C3\ C4\}\{\{*C3\},\{*C2\ *C3\}\}$, where $C3$ is matched at level 1.

c) $\{C5\,C6\}\{\{*C1\},\{*C2\,*C3\}\}$ has no matches at level 0. Hence, a level 1 expansion of the context is taken:

$\{*C5\ C3\ C4\ *C6\}\ \{\{*C1\},\{*C2\ *C3\}\}$ matches nothing at level 1, so a level 2 expansion of the context is taken:

$\{*C5\quad *C3\quad *C4\quad C1\quad C2\quad *C6\}$ $\{\{*C1\},\{*C2\ *C3\}\}$ matches $C1$ OR $C2$ AND $C3$ at level 2. The first covered set is the one to be fired (even though both sets are covered), since it does not have a covered superset. Next, the method of transposition is trivially executed with no resulting change in the logical ordering.

d) Each matched rule antecedent fires a rule consequent, which if not primitive matches exactly one row header (a unique integer) and step (2) iterates.

e) Maintain a count of the maximum number of levels of expansion for the context below the initial level. The GSM is more formally defined as follows: It is given by the existing GSM (i.e., from expanding rule antecedents) plus the maximum number of levels that the context minimally needs to be expanded. This expansion is necessary to get the "first" match using the method of transposition. This stochastic is represented by the maximum depth for any expansion. The GSM and the SSM measure the maximal depth. The algorithm needs to compute these stochastics for each antecedent predicate term so that the best match is fired.

f) If the context fails to be matched, then generalize each term in the starting context one level up in the tree. Remove any redundancies from the resulting generalization. If the generalized context differs from the starting context, then add one to GSM and go to step (5). Otherwise, go to step (6). For example, the starting context $\{C2\,C3\}$ is generalized (one step) to yield $\{C4\ C5\}$. If this now covers a rule antecedent, then the general stochastic measure is one. Otherwise, it is subsequently expanded to yield $\{*C4\ C1\ C2\ *C5\ C3\ C4\}$ at the first level. Notice that the second $C4$ has a longer derivation, is redundant, and would never have been added here. Note too that $C4$ is also a sibling or analog node. If this now covers a rule antecedent, then the general stochastic measure remains one, but the specific stochastic measure is incremented by one to reflect the additional level of specialization.

g) Conflict resolution cannot be a deterministic process as is the case with conventional expert systems. This is because the number of predicates in any match must be balanced against the degree of specialization and/or generalization needed to obtain a match. Thus, a heuristic approach is required. The agenda mechanism will order the rules by their size, general stochastic, and specific stochastic measures with recommended weights of 3, 2, and 1, respectively. Note that this is a heuristic setting and subject to further investigation.

6) Exit the matching process with success (for the entire current context for a row) or failure based on reaching the primitive levels, a timer-interrupt, a forced interrupt, and/or using the maximum allocated memory. A memory or primitive interrupt will invoke step (5f). This enables a creative search until a solution is found or a timer-interrupt occurs. It is perfectly permissible to have a concept appear more than once for reasons of economy of reference, or to minimize the stochastic measures (provide confirming feedback). (Similarly, the preclusion of multiple-inheritance in the predicate trees greatly facilitates maintenance operations, which are inherent to any machine learning process given an open domain.) The stochastic measures also reflect the rapidity with which a concept can be retrieved if not cached.

7) Knowledge acquisition:

a) New rules are added at the head.

b) If exit occurs with failure, or the user deems a selected consequent (in a sequence of consequents) to be in error (trace mode on), then the user navigates the consequent menus to select an attached consequent sequence, which is appropriate for the currently normalized context.

c) If the user deems that the selected "primitive" consequent at this point needs to be rendered more specific, then a new row is opened in the grammar and the user navigates the consequent menus to select a consequent sequence to attach.

d) A consequent sequence can pose a question, which serves to direct the user to enter a more specific context for the next iteration (conversational learning). Questions should usually only add to the context to prevent the possibility of add/delete cycles.

e) Ask the user to eliminate as many specific terms (because the more general terms will tend to match future contexts) from the context as possible (and still properly fire the selected consequent sequence given the assumptions implied by the current row). A context usually consists of a conjunct of terms. This tends to delimit the generality of each term as it contributes to the firing of the consequent. However, once those antecedent terms become fewer in number for use in a subsequent row, then it becomes possible to generalize them while retaining validity. The advantage of generalization is that it greatly increases reusability. Thus, the user needs the capability to substitute a superclass for one or more terms. This implies that perfectly valid entered rules can be replayed with specific (not general) stochastics greater than zero. This is proper, since the specific stochastic preserves validity in theory. Thus, the user may opt to generalize one or more contextual terms by backtracking their derivational paths. If and only if this is the case, step (1) is applied to normalize the result. An undo/redo capability is provided. Validated rule firings are only saved in the rule base if the associated generalization stochastic is greater than zero. The underlying assumption is that rule instances are valid. If a pure rule instance proves to be incorrect, then the incorrect rule needs to be updated or purged and the relevant object class menu(s) may be in need of repair. For example, what is the minimal context to take FIX_CAR to FIX_TIRE? A companion intelligent system could learn to eliminate and otherwise generalize specific terms (randomization theory).

f) The system should verify for the user all the other rule antecedents, in the current row, that would fire, or be fired by the possibly over-generalized context, if matched. (Note that this could lead to a sequence of UNDOs.)

For example, $(\{C5\}, A2)\{(\{C5\}, A1)$ $(\{C5\,C6\}, A2)(\{C5\,C7\}A2, A3)\}$ informs the user that if the new $C5$ acquisition is made, then $A2$ and not $A1$ is proper to fire. If correct, then the result is $\{\{C5\}A2\{C5\,C7\}A2, A3\}$. $\{C5\,C6\}A2$ has been eliminated because it is redundant. Also, $\{C5\ C7\}A2,\ A3$ is fired just in case $C5$ AND $C7$ are true—in which case it represents the most specific selection, since it is a superset of the first set. If the elimination of one or more specific terms causes one or more disjuncts to become proper supersets, then warning message(s) may be issued to enable the user to reflect on the proposed change(s). If the elimination and/or generalization of one or more specific terms enable the firing of another rule in the same row in preference to the generalized rule, then the generalization is rejected as being too general. There is no need to normalize the results, as they would remain in normal form. Besides, any further normalization would counter any necessary speedup.

g) A selected consequent number may not have appeared on the trace path so far with respect to the expansion of each consequent element taken individually. Checking here prevents cycle formation.

h) It should never be necessary to delete the LFU consequent in view of reuse, domain specificity, processor speed, and available memory relative to processor speed. Nevertheless, should memory space become a premium, then a hierarchy of caches should be used to avoid deletions.

8) A metaphorical explanation subsystem can use the antecedent/consequent trees to provide analogs and generalizations for explanative purposes. The antecedent/consequent paths (root, fix_car, fix_tire) serve to explain the recommended action in a way similar to the use of the antecedent and consequent menus. The antecedent/consequent menus will provide disjunction and "user-help" to explain any level of action on the path. The system inherently implements a fuzzy logic known as computing with words [4], [14], [15] (based on the use of conjuncts, descriptive phrases, and tree structures). The virtual rule base is exponentially larger than the real one and only limited by the number of levels in the trees as well as by space-time limitations on breadth-first search imposed by the hardware.

9) A consequent element could be a "do-nothing" element if need be (a Stop Expansion). The provision for a sequence of consequents balances the provision for multiple antecedents. The selected consequent(s) need to be as general class objects as can be to maximize the number of levels and thus the potential for reuse at each level. The consequent grammar is polymorphic, since many such grammars can act (in parallel via the Internet) on a single context with distinct, although complimentary results. Results can be fused as in a multilevel, multicategory associative memory. Multiple context-matched rules may not be expanded in parallel because there can be no way to ascribe probabilities to partially order the competing rules and because any advantage would be lost to an exponential number of context-induced firings. The consequent $\{\ldots\}s$ cannot be ranked by the number of matching terms (for firing the most specific first) because the most specific terms are generally incomparable. However, a covered superset is always more specific than any of its proper subsets. Thus, the first covered set, which does not have a covered superset in the same row, is the one to be fired. If it does have a covered superset, then the superset is fired only if it is the next covered one to be tested in order. It is not appropriate to tag nodes with their level, use a monotonically increasing numbering system, or any equivalent mechanism to prevent the unnecessary breath-first expansion of a node(s) because the menus are dynamic and it would be prohibitively costly to renumber say a terabyte of memory. Node traversal here is not synonymous with node visitation. Even if parallel processors could render the update operation tractable, the search limit would necessarily be set to the depth of the deepest unmatched node. Here, the likelihood of speedup decreases with scale. The contextual terms should only be $*$'d if this does not interfere with their expansion—even if normalized. Let the context be given as $\{C5\,C6\}$ and the RHS be $\{C5\,C7\}$, $\{C1\}$. Clearly, if the context had $*C5$, then the $C1$ might never be matched.

10) Unlike the case for conventional expert systems, a KASER cannot be used to backtrack consequents (goal states) to find multiple candidate antecedents (start states). Also, negation is not used, since the KASER is not a logic in the strictest sense with the attendant advantage of not being subject to essential incompleteness [2]. The problem is that the preimage of a typical goal state cannot be effectively constrained (other than

for the case where the general and specific stochastic are both zero) in as much as the system is qualitatively fuzzy. The answer is to use fuzzy programming in the forward-chained solution [19]. This approach allows the user to enter the constraint knowledge that he/she has into the search. For example, if the antecedent menus are used to specify "CAR" and "FUEL" for the context and the consequent is left unconstrained for the moment, then the system will search through all instances, if any, of CAR crossed with all instances of FUEL (to some limiting depth) to yield a list of fully expanded consequents. Generalization-induced system queries, or consequents that pose questions, if any, will need to be answered to enable this process to proceed. Thus, in view of the large number of contexts that are likely to be generated, all interactive learning mechanisms should be disabled or bypassed whenever fuzzy programming is used. CAR and FUEL are themselves included in the search. Each predicate can also be instantiated as the empty predicate in the case of the antecedent menus, if user-enabled. If the only match occurs for the case of zero conjuncts, then the consequent tree is necessarily empty. A method for fuzzy programming [19], is to simply allow the user to split each conjunct into a set of disjuncts and expand all combinations of these to some fixed depth to obtain a list of contexts. This use of a keyword filter, described below, is optional. For example, the specification $(A \vee A' \vee !A'') \wedge (B \vee B') \wedge (C)$ yields 23 candidate contexts—including the empty predicate (if one assumes that A'' is primitive and allows for redundancy), which excludes the empty context. The exclamation mark, "!" directs the system to expand the nonterminal that follows it to include (in addition to itself) all of the next-level instances of its class. For example, !CAR would yield (CAR TOYOTA FORD MAZDA HONDA $\ldots\lambda$). Here, lambda denotes the empty predicate and is included as a user option.

A capability for expanding to two or more levels if possible ("!!") is deemed to be nonessential, but permissible (for use with relatively few conjuncts). This follows because the combinatorics grow exponentially. One can always take the most successful context(s) produced by a previous trial, expand predicates to another level using "!s" where desired, and rerun the system. Notice that in this manner the user can insert knowledge at each stage—allowing for a far more informed and thus, deeper search than would otherwise be possible. Moreover, the fuzzy specialization engine will stochastically rank the generalized searches to enable an accurate selection among contexts for possible rerun.

The search may be manually terminated by a user interrupt at any time. The search is not to be automatically terminated subsequent to the production of some limit of contexts because to do so would leave a necessarily skewed distribution of contexts—giving the user a false sense of completeness. It is better to have the user enter a manual interrupt and modify the query subsequently. A terminated search means that the user either needs to use a faster computer, or more likely, just narrow down the search space further and resubmit the query.

The user may also have used the consequent menus to specify an optional conjunctive list of key phrases, which must be contained in any generated consequent. Those generated consequents, which contain the appropriate keywords or phrases are presented to the user in rank order—sorted first in order of increasing generalization stochastic and within each level of generalization stochastic in order of increasing specialization stochastic (best-first). For example, (general, specific) $(0,0)(0,1)(1,0)(1,1)\ldots$ Recall that only the specific stochastic preserves validity.

The specified antecedent and consequent classes should be as specific as possible to minimize the search space. Neither the antecedent nor the consequent terms specified by the user are ever generalized. For example, given the consequent class definitions:

$$\begin{aligned}&(\text{COST\_PER\_MILE (CHEAP MODERATE}\\ &\qquad\text{EXPENSIVE)) (MPG(LOW MEDIUM HIGH)).}\end{aligned}$$

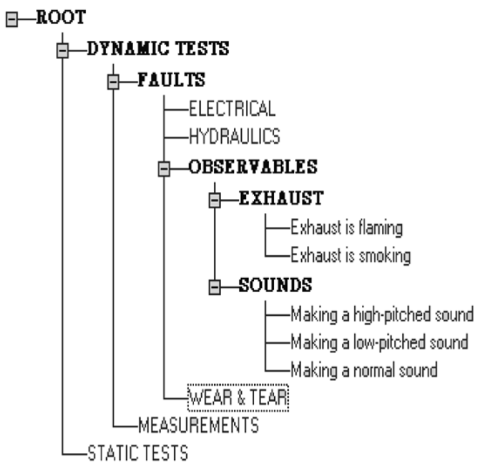
The space of generated consequents can be constrained in a manner similar to the way in which the space of generated antecedents is constrained. Thus, one can write

$$\begin{aligned}&(\text{!CAR}) \wedge (\text{!FUEL}) \wedge (\text{NEW}) \rightarrow\\ &\qquad\qquad (\text{!COST\_PER\_MILE}) \wedge (\text{!MPG}).\end{aligned}$$

This is orthogonal programming: reusing previous paradigms unless there is good reason not to. Here, each candidate solution has been constrained so that it must contain at least one phrase from the four in the COST_PER_MILE class and at least one phrase from the four in the MPG class (including the class name, but excluding the empty predicate). If an asterisk, "$*$" is placed after the arrow, then the compiler is directed not to filter the produced consequents in any way.

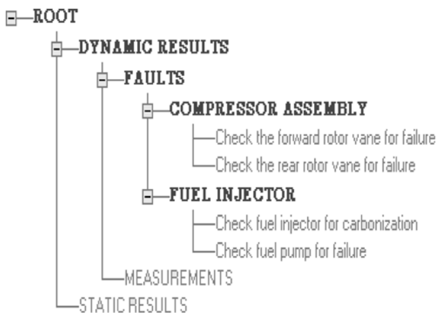
The user can make changes wherever (to the antecedents, the consequents, or both) and whenever (interactively) appropriate and rerun the system query. This represents computing with words because fuzziness occurs at the qualitative level. It is not possible for distinct classes to produce syntactically identical phrases because path names are captured using unique identifiers. That is, the identifiers are always unique even if the represented syntax is not. For example, the concept, "paper clip" would have two unique ids—depending if it terminated a path describing office supplies or electrical conductors. It may help to recall that not all paper clips are conductors of electricity and not all conductors of electricity are office supplies.

It is not deemed necessary to weight the consequent phrases because instance classes preserve validity (at least in theory) and because it would be otherwise impossible to ascribe weights to combinations of words or phrases. For example, "greased" and "lightning" might be synonymous for fast, but taken together



```
ROOT
  DYNAMIC TESTS
    FAULTS
      ELECTRICAL
      HYDRAULICS
      OBSERVABLES
        EXHAUST
          Exhaust is flaming
          Exhaust is smoking
        SOUNDS
          Making a high-pitched sound
          Making a low-pitched sound
          Making a normal sound
      WEAR & TEAR
    MEASUREMENTS
  STATIC TESTS
```

Fig. 2. Screen capture of an antecedent tree.



```
ROOT
  DYNAMIC RESULTS
    FAULTS
      COMPRESSOR ASSEMBLY
        Check the forward rotor vane for failure
        Check the rear rotor vane for failure
      FUEL INJECTOR
        Check fuel injector for carbonization
        Check fuel pump for failure
    MEASUREMENTS
  STATIC RESULTS
```

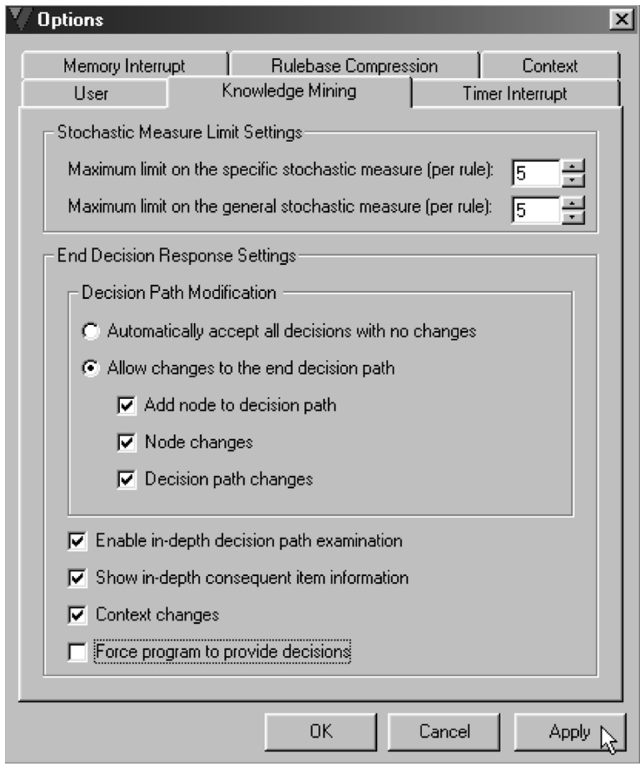
Fig. 3. Screen capture of a consequent tree.

(“greased lightning”), an appropriate weight should be considerably greater than the sum of the partial weights.

To illustrate practical applications of KASERs heuristic search algorithm, we have created the attached KASER.gif, available at http://ieeexplore.ieee.org. It shows two project examples (automotive diagnostics and crystal laser design) and demonstrates how the KASER inference engine uses the predicate trees in the creative reasoning process. Figs. 2–7 (used for discussion purposes in Sections IV and V) are similar to figures included in the video file.

IV. Declarative Object Trees

Object-oriented methods have been used to develop expert systems (e.g., for power management) [20]. Such methods can also facilitate the representation of decision trees [21]. Here, we consider the innovative use of object-oriented methods for conceptual representation. Figs. 2 and 3 depict the declarative object trees that are used to represent knowledge in a KASER. Auto-repair knowledge is depicted here, but since KASER is a shell, there are few limits placed on the operational domain. One limit however is that the more symmetric the domain, the more creative KASER can be. Thus, KASERs possess no inherent advantage over a conventional expert system for learning random historical information; whereas, they can be far more applicable to symmetric domains such as mathematics or chess



Stochastic Measure Limit Settings

Maximum limit on the specific stochastic measure (per rule): 5

Maximum limit on the general stochastic measure (per rule): 5

End Decision Response Settings

Decision Path Modification

○ Automatically accept all decisions with no changes

● Allow changes to the end decision path

☑ Add node to decision path

☑ Node changes

☑ Decision path changes

☑ Enable in-depth decision path examination

☑ Show in-depth consequent item information

☑ Context changes

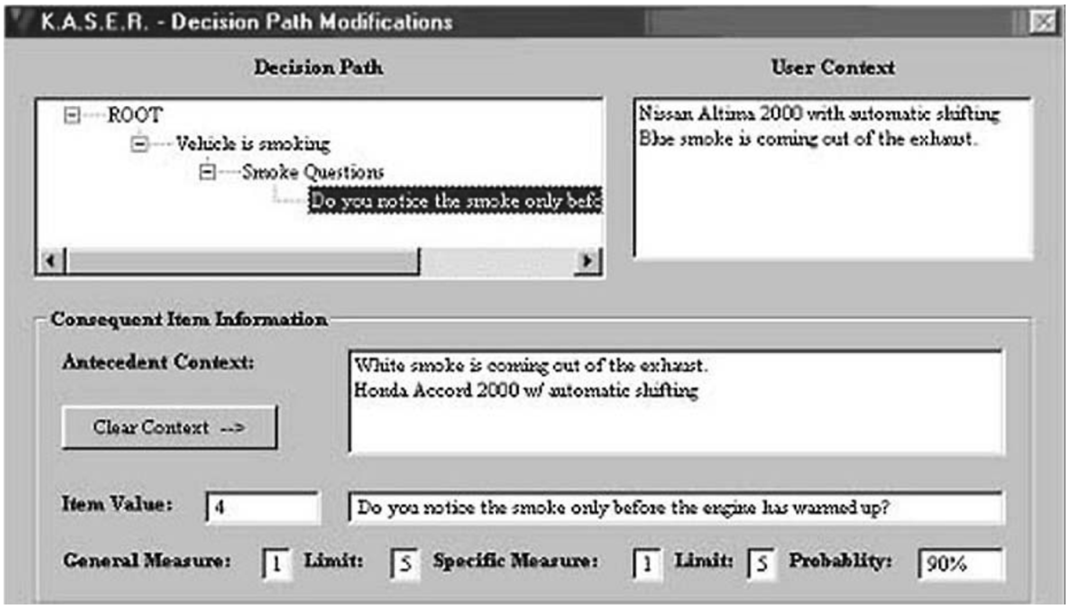
☐ Force program to provide decisions

Fig. 4. Menu for enabling and changing rules.

for example. Here, the system is far more likely to induce correct new knowledge. No nontrivial domain is entirely random or entirely symmetric [22].

Declarative object trees are used as follows. In the antecedent tree, instances fall under an object, while generalizations lie over it. The degree of instantiation or generalization is determined by a tree search, which is limited by a user-defined squelch. A certain quanta of knowledge will lead to a certain object consequent in the consequent tree. Additional knowledge will take you to an instance of that object, if available. For example, the fact that it is cloudy and the barometer is falling will take you to the fact that precipitation is likely. Then, given that it is below freezing, the proper instance of precipitation will be snow. Notice how the use of the consequent object tree maximizes the potential for information reuse. Consequents may be selected as a sequential composition. Objects selected from the antecedent menu serve as the conjunctive context. Note that KASER can also be run on object-oriented functions and procedures. For example, a procedure that enumerates prime numbers is an instance of one that enumerates odd numbers, since every prime number excluding two is necessarily odd.

KASER finds declarative antecedent knowledge that informs the system that the three sounds that an engine might make, subject to dynamic modification, are high-pitched, low-pitched, and normal sounds. By generalizing high-pitched sounds one level to SOUNDS (Fig. 2) and then specializing it one level, one arrives at the first-level analogy—low-pitched sounds, which enables the user context to be matched and leads to the creation of new knowledge. The consequent tree is depicted in Fig. 3 and is similar to the antecedent tree shown in Fig. 2. It is used to generalize rule consequents so as to maximize reusability. Object


K.A.S.E.R. - Decision Path Modifications

Decision Path

ROOT
Vehicle is smoking
Smoke Questions
Do you notice the smoke only befo

User Context

Nissan Altima 2000 with automatic shifting
Blue smoke is coming out of the exhaust.

Consequent Item Information

Antecedent Context:

White smoke is coming out of the exhaust.
Honda Accord 2000 w/ automatic shifting

Item Value: 4 Do you notice the smoke only before the engine has warmed up?

General Measure: 1 Limit: 5 Specific Measure: 1 Limit: 5 Probablity: 90%

Fig. 5. Fuzzy contextual match with resultant decision path.

reuse may simultaneously occur at many levels; however, this example depicts only one level for the sake of clarity. There are many more algorithms, settings, and screens than may or need be detailed in this paper.

Declarative object knowledge is acquired through interaction with a knowledge engineer. This is an on-going process for open domains. Just as one can write a simple program and proceed to evolve it, so too can one evolve the declarative object trees. The more evolved the trees, the more accurate will be the induced knowledge where domain symmetry is held constant. For example, the conceptual object, "apples" may be separated into at least two object subclasses; namely, "red apples" and "green apples"—depending on the application. Then, given an appropriate frame, inductive knowledge pertaining to red apples and its instances is more likely to hold than for green apples and vice versa.

KASER is replete with tools for copying objects, moving objects, pasting (single objects), deep pasting (copying whole subtrees), finding object nodes, suggesting next nodes based on $k = 1$ level of look-ahead, etc. For example, if one is running a preflight checklist, then after checking the rudder the system will look-ahead one-step and suggest checking the elevator. This helps to prevent omissions in complex contextual specifications.

V. Conversational Learning

Limiting the number of levels of induction means limiting system creativity. A limit of zero indicates a disallowance for creativity. For example, in domains such as clinical medicine, one may not want to allow too much creativity lest the possibility of lawsuits will be increased. In artistry, by contrast, one would conceivably allow for unbridled creativity.

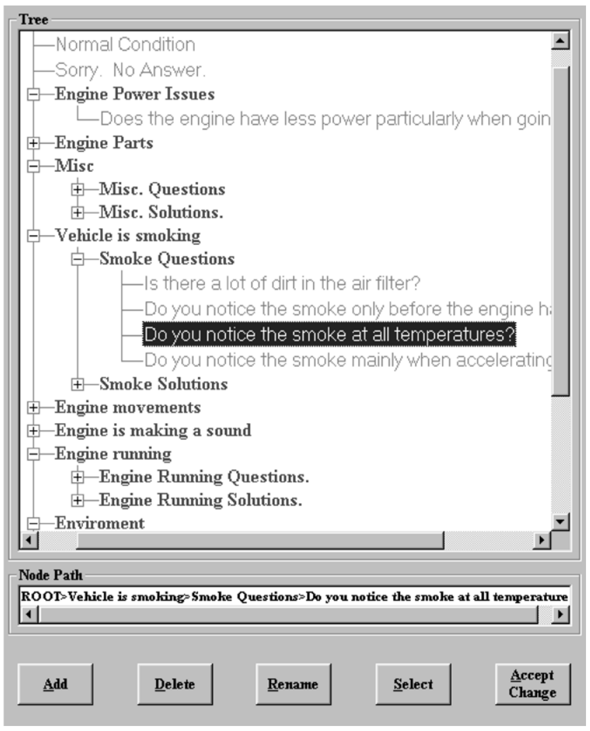
Placing limits on the permitted number of levels of deduction is a practical measure. It can serve to delimit an otherwise combinatorial explosion in the search space. More often, it is used to delimit error creep. That is, while deduction does not introduce error in theory, theory assumes infinitely refined object trees. In as much as this cannot be achieved in practice, deductive operations can introduce error; although, the error introduced here will be far less than that introduced by similar inductive operations. Here, the limit of zero indicates no tolerance for error creep.

Microsoft's "Text-to-Speech" program is provided as royalty-free software for enunciating the context, the answer, and the stochastic measures. The use of a multimodal response helps improve user comprehension and retention of system-generated information.

The Knowledge Mining folder may be selected (Fig. 4) by selecting "Program Options" in the Options menu, where the stochastic squelches may be set. Selecting "Correct Decision" will allow the knowledge engineer to correct erroneous rules (Fig. 5). Selecting "Accept Changes" will streamline the presentation.

Next, we select the context "Nissan Altima 2000 with automatic shifting AND Blue smoke is coming out of the exhaust." In Fig. 5, notice that while there are similarities between the User Context and the matched Antecedent Context, there are also differences. In particular, the user specified a "Nissan Altima 2000 with automatic shifting AND Blue smoke is coming out of the exhaust." The closest matching rule antecedent found was "Honda Accord 2000 with automatic shifting AND White smoke is coming out of the exhaust." An ordered search algorithm found that this qualitatively fuzzy match will minimize the sum of attendant errors and required the use of at most one level of induction and one level of deduction to achieve. The matching predicates serve as a metaphorical explanation for derived consequent actions, if any.

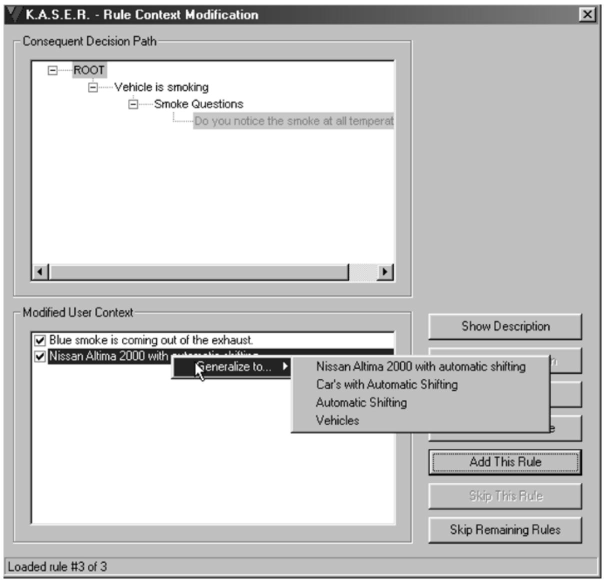
The Decision Path shows the selected hierarchical path in the consequent object tree. Here, we see, "Vehicle is smoking," followed by "Smoke Questions," followed by smoke question instances. Questions are used to help the user complete an otherwise arduous contextual specification. The specified User Context is inducing the presentation of the question "Do you notice the smoke only before the engine has warmed up?"



```
—Normal Condition
—Sorry. No Answer.
⊟—Engine Power Issues
    └—Does the engine have less power particularly when goin
⊞—Engine Parts
⊟—Misc
    ⊞—Misc. Questions
    ⊞—Misc. Solutions.
⊟—Vehicle is smoking
    ⊟—Smoke Questions
        —Is there a lot of dirt in the air filter?
        —Do you notice the smoke only before the engine h
        —Do you notice the smoke at all temperatures?
        —Do you notice the smoke mainly when acceleratinc
    ⊞—Smoke Solutions
⊞—Engine movements
⊞—Engine is making a sound
⊟—Engine running
    ⊞—Engine Running Questions.
    ⊞—Engine Running Solutions.
⊟—Enviroment
```

Node Path
ROOT>Vehicle is smoking>Smoke Questions>Do you notice the smoke at all temperature

Fig. 6. Correcting an erroneous question node.



```
⊟—ROOT
    ⊟—Vehicle is smoking
        ⊟—Smoke Questions
            └—Do you notice the smoke at all temperat
```

☑ Blue smoke is coming out of the exhaust.
☑ Nissan Altima 2000 with

Nissan Altima 2000 with automatic shifting
Car's with Automatic Shifting
Automatic Shifting
Vehicles

Loaded rule #3 of 3

Fig. 7. Rule context modification window.

The Change Node button permits one to change the question presented. New questions or declarative statements can always be inserted through the use of the Add button. The Clear Context button replaces the Antecedent Context with the User Context so as to create a new user rule.

The Add Node button permits one to add a more specific node to the decision path. The Change Decision button permits one to replace the selected node with any other consequent node. The Delete Node button will delete the selected node along with all of its children. The Expand All and Collapse All buttons are used to expand and contract the decision path respectively. The function of the Show End Path (Show Entire Path) buttons is similar, where the end path refers to the leaf nodes. The Description button provides a pop-up window that contains a textual description of the node, if present. The functionality of the remaining buttons is relatively self-descriptive. Clicking on the Change Node button in Fig. 5 results in bringing up the consequent tree depicted in Fig. 6. Here, the question is being changed from, "Do you notice the smoke only before the engine has warmed up?" to "Do you notice the smoke at all temperatures?"

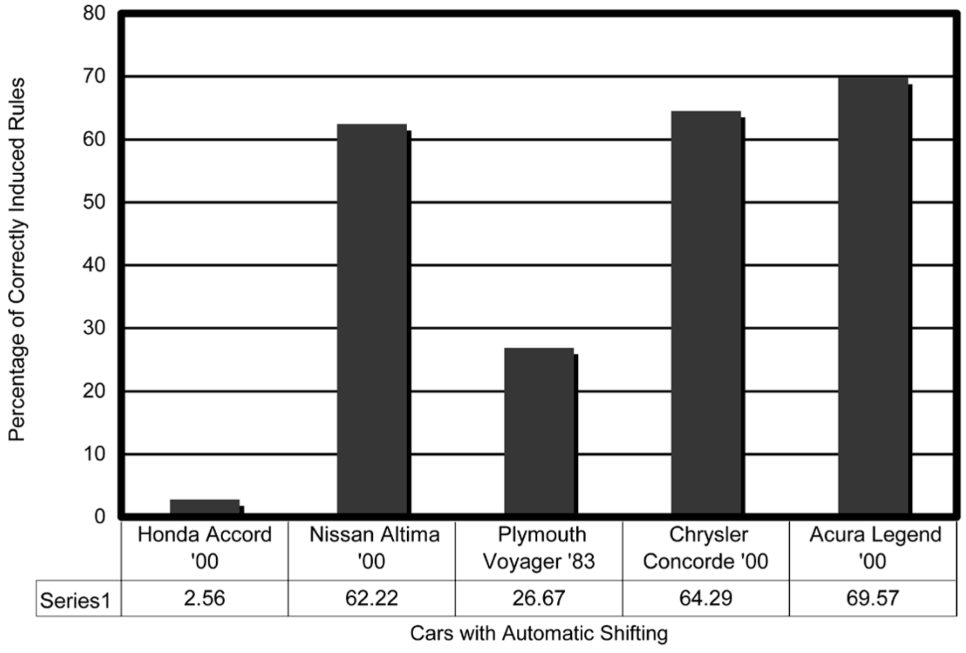
The Rule Context Modification Window (Fig. 7) allows the knowledge engineer to generate a most general antecedent to be paired with the rule consequent. A more general antecedent implies a greater potential for reuse.

Contextual predicates are generalized through two means. First, unchecking a contextual predicate will remove it from the antecedent. The resulting more general rule may then be checked for contradiction with the existing rule base by clicking on the Check Current Rule button, which is hidden in Fig. 7. Similarly, the knowledge engineer may also generalize an included predicate by right clicking on it and selecting one from among the presented hierarchy of generalizations, if any. Again, the resulting more general rule may then be checked for contradiction with the existing rule base as before.

Resulting rules can be acquired by clicking on the Add This Rule button. Clicking on the Skip This Rule button will prevent its acquisition. In this case, one relies on a previously existing rule to navigate to the next level, or the transition will not be possible. Clicking on the Skip Remaining Rules button will preclude the acquisition of any rules here. Using the Add This Rule button, one can add a different rule (including different generalizations) to effect the transition level by level in the tree (precipitation to snow). Again, this maximizes the potential reuse of knowledge because distinct rules can operate at distinct levels in the consequent object hierarchy. This is a key advantage associated with the use of the declarative object trees.

The context, "Nissan Altima 2000 with automatic shifting AND Blue smoke is coming out of the exhaust," is appended with the information that the "smoke is noticed at all temperatures." One level of generalization and specialization are minimally needed to match the resultant context against the present rule base. The answer that results is that there is a Blown Head Gasket. Notice that this diagnosis is not available through the knowledge base or real rule space and is open under strict deduction. Rather, it is synthesized at runtime through the creation of a much larger virtual rule space.

The General Stochastic Measure shows that no less than one level of inductive inference was needed to arrive at the given answer. The answer was correct 90% of the time to date that the process for arriving at it included this GSM. The probability of error here is dependent on the size of the existing knowledge base at the time as well as the degree of domain symmetry (Fig. 8). Erroneous rules require supervised correction. A property of KASER is that the effects of any correction propagate supra-linearly and bear proportion to the degree of domain symmetry. Fig. 8 illustrates this claim based on empirical evidence.



	Honda Accord '00	Nissan Altima '00	Plymouth Voyager '83	Chrysler Concorde '00	Acura Legend '00
Series1	2.56	62.22	26.67	64.29	69.57

Fig. 8. Knowledge transference and supra-linear learning in KASER applied to automotive diagnostics.

Empty buttons allow for the specification of different answers. The Accept and Delete buttons are used to complete the specification.

KASER, as implemented, includes application-oriented issues. Pictures, Sound, Video, and Text may be attached to an antecedent or consequent object. These attachments serve to enhance user understanding, memory, and most importantly, serve to define otherwise poorly understood issues. For example, it may not be clear that an automobile under discussion is a Honda Accord 2000 with Automatic Shifting. By right clicking on this object, one can bring up a picture of it and verify the car model.

Antecedent and consequent predicates can be attached with such picture multimedia as .bmp, .dib, .emf, .gif, .ico, .jpg, and .wmf. Sound files such as .mp3 and .wav and video files such as .avi, .mov, and .mpg may also be attached to objects. The Description option allows the knowledge engineer to attach textual descriptions to predicates. Any predicate can have all or no multimedia attached.

Again, subtrees can be Copied and Deleted through the use of the appropriate commands. Individual nodes can be Pasted and entire subtrees can be Deep Pasted to wherever the mouse has been set.

The Move command provides for the movement of entire subtrees to different locations. If the Shift or Ctrl keys are used, then directed subclasses might be moved. For example, in an object list of apple types (Granny, McIntosh, Red Delicious, *et al.*), one could easily move all of the red ones and all of the green ones into separate object menus under the apple class. KASER would then treat red and green apples as distinct classes where necessary for rule induction and/or deduction. Clearly, the provision for such exacting and evolving class formation provides the knowledge engineer with a powerful capability. That is, KASER can learn to absorb many of the nuances of domain-specific human creative thought. This is something that current neural architectures (those that compute nonlinear discriminants through the use of hidden layers) can never do [23], [24].

Fig. 8 provides a graphical depiction of empirical KASER learning results for the domain of automotive diagnostics. The most stringent test of learning occurs where the GSM is greater than zero and the yielded consequent is deemed to be correct. In other words, the validity of virtual rules arrived at by inductive processes is subject to investigation.

KASER was "cold-started" for the Honda Accord. Rule (next to fix columns, words go to next column rather than below figure) acquisitions are performed to correct wrong answers. Here, only 2.56% of the 50 contexts presented to the system, which resulted in a $\mathrm{GSM} > 0$ were deemed to yield the correct consequent by an oracle [25]. The poor performance is attributed to the sparsely populated knowledge base. We chose the Nissan for our next automobile because it is quite similar to a Honda in all of its salient features. The rule base at this point consists of 141 Honda rules. We proceeded to train the system on Nissan rules. KASER gave the correct consequent for 62.22% of the Nissan contexts, which resulted in a $\mathrm{GSM} > 0$. This striking improvement in performance may be attributed to knowledge transference from the Honda rules. In all, 120 rules were supplied for the Nissan.

Next, a vehicle that is relatively random when compared to the Honda and the Nissan was chosen. Such a vehicle is the Plymouth Voyager. Experiments revealed that only 26.67% of the Voyager contexts, which resulted in a $\mathrm{GSM} > 0$ gave the correct consequent. In all, 164 rules were supplied for the Voyager. The drop in performance was to be expected and may be attributed to the relative lack of symmetry between the Voyager and the two cars. However, note that the percent age correct is still greater than that for the Honda due to transference.

Next, KASER gets 64.29% of the Concorde contexts, which resulted in a $\mathrm{GSM} > 0$, correct. The Chrysler Concorde is similar to the other cars. In all, 161 rules were supplied for the Concorde. Here, the excess of 2% improvement in the percentage correct over the Nissan may be attributed to transference from the Voyager.

Finally, the Acura Legend, which is similar to all the other cars, gets 69.57% of the Acura contexts, which resulted in a $\mathrm{GSM} > 0$, correct. In all, 154 rules were supplied for the Acura. Here, the improvement may be attributed to domain symmetry. KASER clearly has not lost its knowledge of cars and appears to be approaching an asymptote for inductive accuracy. Once the knowledge base has been populated (as shown), this asymptote is more a function of the degree of inherent domain symmetry. The supra-linear propagation of corrections is evident from the fact that knowledge gained from a single domain transfers to a plethora of similar domains [26]. Clearly, as demonstrated by Fig. 8, learning about cars facilitates learning about vans and vice versa.

VI. Conclusions

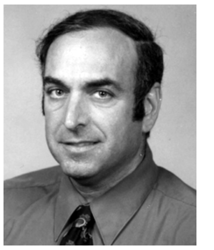
Processing knowledge is abstract and dynamic. As future knowledge management applications attempt to mimic the human decision-making process, a language(s) is needed which can provide developers with the necessary tools to achieve these goals.

In conclusion, the knowledge-acquisition bottleneck has been and will continue to be broken. KASER offers itself as a novel and more creative type of expert system. This third-generation expert system computes with words and employs qualitative fuzzy reasoning. It also may be said to fail softly and for this reason is not brittle. KASER's cannot be realized through the exclusive use of predicate logic as they embed an inductive component. Furthermore, KASER's can in principle capture and otherwise represent virtually all of the domain knowledge possessed by a knowledge engineer. They can also predict their own likelihood of error through incorporation of a simply history function. They reason through a gradient descent or ordered search algorithm, which serves to minimize this error.

KASER has exhibited domain transference and supra-linear learning, which bear proportion to the inherent degree of domain symmetry. Finally, it introduces the use of single inheritance declarative object trees in expert systems. Such trees facilitate the capture of object-oriented semantic relations among rule predicates and serve the processes of metaphorical explanation as a consequence.

References

[1] G. J. Chaitin, "Randomness and mathematical proof," *Sci. Amer.*, vol. 232, no. 5, pp. 47–52, 1975.
[2] V. A. Uspenskii, *Gödel's Incompleteness Theorem, Translated From Russian*. Moscow, Russia: Ves Mir, 1987.
[3] J.-H. Lin and J. S. Vitter, "Complexity results on learning by neural nets," *Mach. Learn.*, vol. 6, no. 3, pp. 211–230, 1991.
[4] S. H. Rubin, "Computing with words," *IEEE Trans. Syst. Man, Cybern. B*, vol. 29, pp. 518–524, Aug. 1999.
[5] E. A. Feigenbaum and P. McCorduck, *The Fifth Generation*. Reading, MA: Addison-Wesley, 1983.
[6] S. H. Rubin, "The role of computational intelligence in the new millenium," in *Proc. Plenary Speech 3rd World Multiconference Systemics, Cybernetics, Informatics (SCI) 5th Int. Conf. Information Systems Analysis Synthesis (ISAS)*, Orlando, FL, July 1999, pp. 3–13.
[7] S. H. Rubin, L. Trajkovic, J. Boerke, and R. J. Rush Jr, "On the role of randomization in software engineering," in *Proc. Intern. Conf. Industry, Engineering, Management Systems*, 2001, pp. 78–83.
[8] M. Minsky, *The Society of Mind*. New York: Simon and Schuster, 1987.
[9] C. T. Clark, "An interview with Marvin Minsky," *Knowl. Manage.*, pp. 26–28, June 2000.
[10] S. Amarel, "On representations of problems of reasoning about actions," *Mach. Intell.*, vol. 3, pp. 131–171, 1968.
[11] A. Roy, "On connectionism, rule extraction, and brain-like learning," *IEEE Trans. Fuzzy Syst.*, vol. 8, pp. 222–227, Apr. 2000.
[12] A. K. C. Wong and W. Yang, "Pattern discovery: A data driven approach to decision support," *IEEE Trans. Syst., Man, Cybern. C*, vol. 33, pp. 114–124, Feb. 2003.
[13] L. A. Zadeh, "From computing with numbers to computing with words—From manipulation of measurements to manipulation of perceptions," *IEEE Trans. Circuits Syst.*, vol. 46, pp. 105–119, Jan. 1999.
[14] S. H. Rubin, R. J. Rush Jr, J. Boerke, and L. Trajkovic, "On the role of informed search in veristic computing," in *Proc. 2001 IEEE Int. Conf. Systems, Man, Cybernetics*, 2001, pp. 2301–2308.
[15] S. H. Rubin, R. J. Rush, Jr., J. Murthy, M. H. Smith, and L. Trajkovic, "KASER: A qualitatively fuzzy object-oriented inference engine," in *Proc. North American Fuzzy Information Soc.*, 2002, pp. 354–359.
[16] A. Barr and E. A. Feigenbaum, *The Handbook of Artificial Intelligence*. Los Altos, CA: William Kaufmann, 1981, vol. 1.
[17] ——, *The Handbook of Artificial Intelligence*. Reading, MA: Addison-Wesley, 1982, vol. 2.
[18] H. M. Meng and K. C. Siu, "Semiautomatic acquisition of semantic structures for understanding domain-specific natural language queries," *IEEE Trans. Knowledge Data Eng.*, vol. 14, pp. 172–181, Jan./Feb. 2002.
[19] S. H. Rubin, "A fuzzy approach toward inferential data mining," *Comput. Ind. Eng.*, vol. 35, no. 1–2, pp. 267–270, 1998.
[20] Q. Liu and S. Cheng, "Object-oriented methods drive protective relay system," *IEEE Comput. Appl. Power*, vol. 13, pp. 33–37, Jan. 2000.
[21] N. R. Pal and S. Chakraborty, "Fuzzy rule extraction from ID3-type decision trees for real data," *IEEE Trans. Syst., Man, Cybern. B*, vol. 31, pp. 745–754, Oct. 2001.
[22] S. H. Rubin, "New knowledge for old using the crystal learning lamp," in *Proc. IEEE Int. Conf. Systems, Man, Cybernetics*, 1993, pp. 119–124.
[23] K. Chellapilla and D. B. Fogel, "Evolution, neural networks, games, and intelligence," *Proc. IEEE*, vol. 87, pp. 1471–1496, Sept. 1999.
[24] S. H. Rubin and L. Trajkovic, "On the role of randomization in minimizing neural entropy," in *Proc. 5th Multi-Conf. Systemics, Cybernetics, Informatics*, vol. XV, 2001, pp. 235–240.
[25] "Software to Help You Troubleshoot Your Car, Truck, Van, or SUV 1983–2001," Motor Trend AutoTech., EMAP, 2001.
[26] S. H. Rubin, R. J. Rush Jr, M. H. Smith, S. N. J. Murthy, and L. Trajkovic, "A soft expert system for the creative exploration of first principles of crystal-laser design," in *Proc. 2002 IEEE Int. Conf. Systems, Man, Cybernetics*, 2002, pp. TA203.1–7.
[27] Y. Dote and S. J. Ovaska, "Industrial applications of soft computing: A review," *Proc. IEEE*, vol. 89, pp. 1243–1265, Sept. 2001.
[28] A. J. Kfoury, R. N. Moll, and M. A. Arbib, *A Programming Approach to Computability*. New York: Springer-Verlag, 1982.
[29] M. A. Arbib, A. J. Kfoury, and R. N. Moll, *A Basis for Theoretical Computer Science*. New York: Springer-Verlag, 1981.
[30] S. H. Rubin, "Knowledge Amplification by Structured Expert Randomization (KASER), Assigned to the U.S. Navy and Filed With the USPTO," U.S. Patent NC 83014, July 24, 2002.



Stuart H. Rubin (M'88–SM'00) received the B.S. degree in business from the University of Rhode Island, Kingston, the M.S. degree in industrial and systems engineering from Ohio University, Athens, the M.S. degree in computer science from Rutgers University, Piscataway, NJ, and the Ph.D. degree in computer and information science from Lehigh University, Bethlehem, PA, in 1975, 1977, 1980, and 1988, respectively.

He is a Senior Scientist at the Space and Naval Warfare Systems Center (SSC), San Diego, CA, code 2734. He has authored over 140 refereed conference and journal papers, as well as several patent applications on behalf of SSC, San Diego. His professional interests center about heuristic methodologies for multisensor fusion, decision support, and knowledge discovery. He is Associate Editor for the *International Journal of Modeling and Simulation* and the *Journal of Systemics, Cybernetics, and Informatics*. He has delivered several keynote lectures at international conferences.

Dr. Rubin was awarded the IEEE Systems, Man, and Cybernetics Society's Outstanding Service Award in 2003. He is a member of the American Association for the Advancement of Science (AFCEA), the New York Academy of Sciences, the North American Fuzzy Information Processing Society, and several other scientific societies. He chairs the IEEE Systems, Man, and Cybernetics Technical Committee on Knowledge Acquisition in Intelligent Systems. He is also Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, C. He is the Founder and General Chair of the IEEE International Conference on Information Reuse and Integration (IRI). He also currently serves on the IEEE Advisory Committee.

S. N. Jayaram Murthy (M'75) received the B.E. degree in electronics and communications from the Engineering College, Kakinada, India, in 1967, the M.Tech. degree in electrical engineering from the Indian Institute of Technology, Kanpur, India, in 1969, and the Ph.D. degree in computer science from the University of Illinois, Urbana-Champaign, in 1974 for his work on visual texture analysis and synthesis.

He is currently a Professor in the Department of Computer Science, Central Michigan University (CMU), Mt. Pleasant. He also served as the Chairperson of the Computer Science Department at CMU for five years. Before joining CMU, he worked as a regular faculty member at the following institutions: the University of South Carolina, Columbia, Wayne State University, Detroit, MI, and the Indian Institute of Science, Bangalore, India. His publications, research, and teaching interests are in the areas of data/text mining, multimedia, artificial intelligence, computer vision, pattern recognition, and image processing. He served as a Co-Chair of the Information Reuse and Integration (IRI) conference that was held in Honolulu, HI in Nov. 2001.

Dr. Murthy is also a member of the Association of Computing Machinery (ACM), the Society of Computer Simulation (SCS) International, and the International Society for Computers and their Applications (ISCA). He has been a Member of the international technical program committee for all the IRI conferences since 1997.

Michael H. Smith (SM'99) received the B.S. degree in engineering, the M.S. degrees in engineering and business administration, and the Ph.D. degree in electrical engineering and computer science from the University of California, Berkeley, in 1973, 1976, and 1994, respectively.

He has an extensive business background and his research interests are in fuzzy systems with application to intelligent control, robotics, knowledge acquisition, and business systems.

Dr. Smith received the IEEE Systems, Man, and Cybernetics Society's Outstanding Contribution Award in 1998. In 1996, he was given the North American Fuzzy Information Processing Society K. S. Fu Award. He is currently the Jr. Past-President of the IEEE Systems, Man, and Cybernetics (SMC) Society, and has previously held various positions with SMC: President, Vice-President, Long Range Planning and Finance, Treasurer, member of the Ad-Com, Cybernetics Area Coordinating Chair, Chair of the Technical Committee on Computational Intelligence, and SMC representative to the Neural Network Council. Furthermore, he has served on the organizing committees of a number of SMC conferences and has organized a large number of invited sessions for these conferences. He has also served on the IEEE Technical Activities Board. He is a Past-President of the North American Fuzzy Information Processing Society, and a past Vice-President of the International Fuzzy Systems Association. He was a Co-program chair for the International Conference on Intelligent Processing and Manufacturing of Materials in 1997 and 1999. He was the General Chair for NAFIPS'96, and was also the Co-General Chair of the Joint 9th IFSA World Congress and 20th NAFIPS International Conference in 2001.

Ljiljana Trajković (S'78–M'86–SM'95) received the Dipl.Ing. degree in her native Yugoslavia in 1974, the M.Sc. degrees in electrical engineering and computer engineering from Syracuse University, Syracuse, NY, in 1979 and 1981, respectively, and the Ph.D. degree in electrical engineering from University of California, Los Angeles, in 1986.

She is currently a Professor in the School of Engineering Science at Simon Fraser University, Burnaby, BC, Canada. From 1995 to 1997, she was an NSF Visiting Professor in the Electrical Engineering and Computer Sciences Department at University of California, Berkeley. She was a Research Scientist at Bell Communications Research, Morristown, NJ, from 1990 to 1997, and a Member of Technical Staff at AT&T Bell Laboratories, Murray Hill, NJ, from 1988 to 1990. Her research interests include high-performance communication networks, intelligent control of communication systems, computer-aided circuit analysis and design, and theory of nonlinear circuits and dynamical systems.

She is Technical Program Co-Chair of ISCAS 2005. She served as Technical Program Chair and Vice General Co-Chair of ISCAS 2004. Dr. Trajkovic is currently serving on the Board of Governors of the IEEE Circuits and Systems Society and the IEEE Systems, Man, and Cybernetics Society. She is an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS (PART I). She was an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS (PART II) from 2001 to 2003, the IEEE CIRCUITS AND SYSTEMS MAGAZINE from 2001 to 2003, and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS (PART I) from 1993 to 1995. She was Chair of the IEEE Technical Committee on Nonlinear Circuits and Systems in 1998. She is currently Chair of the IEEE Circuits and Systems Society joint Chapter of the Vancouver/Victoria Sections.