Knowledge Acquisition for Performance Systems; or: When can "Tests" Replace "Tasks"?

Tim Menzies, Paul Compton

Artificial Intelligence Laboratory,
School of Computer Science and Engineering,
University of New South Wales, PO Box 1, Kensington, NSW, Australia, 2033
{timm|compton}@cse.unsw.edu.au

ABSTRACT

Currently, "task analysis" is the dominant paradigm in the knowledge acquisition community. We argue that for *performance systems* (i.e. systems that do not have to offer a knowledge-level description of their performance at runtime) a simpler "test analysis" approach may suffice. We offer examples were a seemingly-naive testing regime gives rise to competent performance systems. Further, by certain measures, these systems developed via test analysis out-performed systems developed for similar domains using other techniques. Test analysis did not augment some other methodological approach: it removed the need for any other methodology. We speculate that for performance systems, task analysis could be deferred till after the development of a tested performance system. That is, for performance systems, testing replaced task but task analysis could augment test analysis once a system was in production.

1. INTRODUCTION

Task analysis evolved from a reverse engineering of existing expert systems. We use the term "task analysis" to group together a set of researchers who argue for a similar structure using different terminology; e.g. KADS (Wielinga, Schreiber et al. 1992), Chandrasekaran's generic tasks (Chandrasekaran 1986), Steels' components of expertise (Steels 1990), Clancey's model construction operators (Clancey 1992), and the architecture of SPARK/BURN/FIREFIGHTER (Marques, Klinker et al. 1991). Currently, task analysis is the dominant paradigm in the knowledge acquisition (KA) community[1].

Here we present "test analysis", our own reverse engineering of the ripple-down-rules (RDR) and hypothesis testing approaches. Both systems had certain similarities:

---

[1] Section 2.1 gives an overview of task analysis but the precise details of this approach are beyond the scope of this paper. The interested reader is referred to the "related work" section of (Wielinga, Schreiber et al. 1992) for an historical overview and a comparison of the different approaches. The dominant task analysis technique is KADS. For a gentle introduction to KADS, see the short tutorial in (Linster and Musen 1992) followed by (Wielinga, Schreiber et al. 1992). For a detailed introduction to the technique, see (Tansley and Hayball 1993).

Design: An approach to KB construction was proposed that relied on low-level representation primitives.

Use of Test Cases: Test cases were used during the development to test the knowledge and control the development.

Traces: The systems could offer support for detecting and fixing KB anomalies only in terms of low-level rule-traces.

Test Domains: The target domain was complex: interpreting biochemical results and modelling neuroendocrinological theories.

Of Course it Won't Work: The pre-experimental intuition was that the proposed method was obviously naive and doomed to failure. Previous research in the same domains relied extensively on high-level abstracted constructs.

Oh… It Worked: Once implemented, the systems displayed adequate competency in their target domains. In fact, by certain measures, the systems, based on seemingly-naive representations, out-performed existing systems based on more complex representations.

Maintenance Tools: Tools for maintaining the system were developed as part of the initial product. Test analysis does not divide development into implementation then maintenance: tools for maintaining the knowledge have to be developed prior to commencing KA. Consequently, test analysis permits the seamless extension of the initial system into the maintenance system.

This experience leads us to speculate that using test analysis, we could create competent expert systems which (i) can complete their inferencing in reasonable time; (ii) can exhibit satisfying competency in their target domain; and (iii) provide sufficient insight into the inner workings of the knowledge base to detect and fix KB anomalies[2] during the development and maintenance cycle. Further, if the goal of the system is not (i) a teaching tool or (ii) a theoretical tool for generalising old designs, then this performance system generated via test analysis would suffice and the overheads and problems of task analysis could be avoided.

The structure of this paper is as follows. Section two gives an overview of the strengths and weaknesses of the task paradigm and defines *performance systems*. Sections three and four describe our test analysis systems. Section five debates certain objections to our proposal that we have encountered in the past.

2. TASK ANALYSIS

2.1. OVERVIEW

2.1.1 A reverse engineering of existing expert systems has revealed that above the level of rules, frames, forward/ backward chaining, etc., there exist repeated meta-level patterns of inference (Chandrasekaran 1983; Clancey 1985; Kahn, Nowlan et al. 1985; Van de Brug, Bachant et al. 1986; Clancey 1992).

---

[2] A KB anomaly is a mismatch between the expectation of the domain expert and what can be found either in the KB or in the output of the KB's execution. Note that we do not use the term "error" since this requires assumptions of Platonic "truth" or "falsehood" which we reject. This point is discussed further in the section 5.3. "Incorrect Models".

2.1.2 Software that stores parameterised versions of these meta-level patterns offers a structured approach to the knowledge acquisition endeavour (Chandrasekaran 1986; Chandrasekaran 1990; Steels 1990; Clancey 1992; Wielinga, Schreiber et al. 1992). KA can initially be the hunt for a match between known expert behaviour and known meta-level patterns (Wielinga, Schreiber et al. 1992). Once a match is found, then the meta-level pattern offers a rich description of the high-level processing loops as well as the data structures that the experts have to supply.

2.1.3 Within the meta-level patterns, there often exist processing modules used in other meta-level patterns (e.g. "classify", "select", "match", "generalise", etc) (Chandrasekaran 1986; Marques, Klinker et al. 1991; Clancey 1992; Wielinga, Schreiber et al. 1992). Software libraries that support these processing modules can be used to decrease development time. Marques *et al* compare development times with and without such a software library: 200 days (without) to 20 days (with) (Marques, Klinker et al. 1991).

2.1.4 Some evidence suggests that encoding knowledge bases (KBs) using these meta-level patterns gives the inference engine enough control knowledge to significantly improve its performance. Clancey reports that in one task-analysis reverse engineering of MYCIN, all the search was removed from the inferencing (Clancey 1992). This would be consistent with the theory that novices spend much of their inferencing time working out what to do next (i.e. setting goals) while experts already know the relevant steps in their inferencing and so reach their conclusions faster (Larkin, McDermott et al. 1980).

2.1.5 Explanations generated from the meta-level patterns are more insightful for the domain expert and the software designer than a (e.g.) low level trace of rule firings (Clancey 1983; Wielinga, Schreiber et al. 1992). Such explanations can be used to identify and fix knowledge anomalies during KA.

2.1.6 These explanations can also be used to offer justifications of the delivered system's conclusions for the end-user.

2.1.7 Despite early attempts to formalise the expert systems construction process (Stefik, Aikins et al. 1982), expert system construction remains a somewhat hit-and-miss affair. By the end of the 1980s, it was recognised that our design concepts for expert systems were incomplete (Buchanan and Smith 1989). Task analysis provides us with a rich and succinct vocabulary for describing, summarising, and comparing expert systems. For example, a KADS-level description of heuristic classification fills 10 pages (Akkermans, van Harmelen et al. 1992) while Clancey's original description is somewhat more verbose (61 pages) and not as precise (Clancey 1985). Practitioners find this retrospective second-glance at their systems useful for developing more generalised architectures for future work (Linster and Musen 1992). Expert systems theoreticians can use task analysis to assess and clarify the essential features and differences of applications (Schreiber, Wielinga et al. 1992). Lastly, knowledge engineering novices can use a task analysis of classic expert systems to quickly review successful techniques.

2.2. PERFORMANCE SYSTEMS

Note that, with the exception of point 2.1.7 above, the primary advantages of task analysis is to reduce the complexity and (hence) time required for the construction of *performance expert systems*; i.e. a system that must (i) achieve some goal at runtime and (ii) provide assistance at

design time for KA. Pure performance systems cannot do point seven above; i.e. deliver a knowledge-level (Newell 1982) description of their competency or reasons for their conclusions.

A performance system need not have tools for browsing its knowledge base, provided that some design time support was available for verifying the semantics of this black-box knowledge base. While these verification tools have to present a summary of the KB to the domain expert/ knowledge engineer, this summary need not be a knowledge-level description (for example, see the discussion in section 3.3. on the traces from the RDR system).

2.3. PROBLEMS WITH TASKS

In this section we describe the overheads and problems associated with task analysis.

2.3.1 Task analysis is an active research area and mature tools for this technique are still evolving. Even simple documentation tools are lacking[3].

2.3.2 Between the various camps of task researchers, there is little agreement on the details of the precise structure of the meta-level patterns. Contrast the list of repeated processing modules from KADS (Wielinga, Schreiber et al. 1992) and SPARK/ BURN/ FIREFIGHTER (Marques, Klinker et al. 1991) (termed "knowledge sources" and "mechanisms" respectively). While there is some overlap, the lists are different. Further, the number and nature of the meta-level patterns is not fixed. Often when a domain is analysed using tasks, a new meta-level pattern is induced (Tu, Shahar et al. 1991; Linster and Musen 1992).

2.3.3 The existing meta-level patterns may not be correct. Consider, as one example, the KADS *interpretation model*[4] for diagnosis (Wielinga, Schreiber et al. 1992). Our reading of the model-based diagnosis (MBD) literature (Hamscher, Console et al. 1992) is that the KADS diagnosis interpretation models ignore many important features of the research into diagnosis. For example, the interpretation models listed in (Tansley and Hayball 1993) separate diagnosis from repair; a questionable division.

We make no comment here as to the correctness of (e.g.) the KADS diagnosis model. We only note that the knowledge engineering fields that task analysis is trying to generalise are by no means "closed-books" but are themselves subject to rapid evolution[5]. Such an evolution could imply fundamental redesigns of the meta-level patterns. For example: recent work on abduction[6] suggests that a wide variety of knowledge engineering tasks can be usefully described as different forms of abduction (O'Rourke 1990). General KR architectures for a variety of knowledge-level tasks could be implemented on top of an abductive inference system. This approach would have at least two advantages: (i) a common underlying representation would facilitate knowledge

---

3 Prior to the publication of (Tansley and Hayball 1993) there existed no central site, or even a Internet FAQ list of commonly used models. Further, many of the models listed in (Tansley and Hayball 1993) were created especially for the book by the authors from their own undocumented sources (see paragraph 4, page 260).

4 KADS-speak for a meta-level inference pattern.

5 An informal KADS seminar convened at DX '93 (Hewlett-Packard 1993) concluded that the KADS interpretation models for diagnosis were premature generalisations of a developing field.

6 Given some rule $\alpha \rightarrow \beta$, then *deduction* means concluding β given α and *abduction* means concluding α given β. *Induction* means concluding $\forall X \alpha(X) \rightarrow \beta(X)$ from the example $\alpha \rightarrow \beta \wedge \alpha \wedge \beta$

sharing; (ii) formal results regarding the tractability of abduction could be used by the knowledge engineer to avoid the acquisition of knowledge that is intractable to process[7].

2.3.4 The bottleneck in task analysis is the mapping between problem description and meta-level inference patterns. In the SPARK/ BURN/ FIREFIGHTER work, for example, the application domains were all pre-selected and the mappings between a library of application types and the library of mechanisms was hand-coded. Techniques to automate this process have yet to evolve.

2.3.5. One technique we have found useful for expert system development is prototyping (Menzies, Black et al. 1992). The overheads associated with documenting (e.g.) KADS may introduce an organisational inertia inhibiting the prototyping process ("you mean you want we to re-write the design document... again?")[8].

2.3.6 Any reading of the task analysis literature suggests that one requires a high level of skill to use this approach (perhaps because of points 2.3.1 to 2.3.5). Allemang note that "generic task analysis" (the Chandrasekaran school of task analysis) is difficult and requires a knowledge engineer (Allemang 1992). Marques *et al* similarly note that experts are enmeshed in the details of using their skills and find it difficult to understand what they are doing in more abstract terms (Marques, Klinker et al. 1991). Aben provides lists of authors who express discontent with imprecisions in the KADS formalism (Aben 1992).

2.3.7 The problem of explanation is not solved merely by offering a trace of the system's traversal over a task description (e.g. a KADS interpretation model). The issue of appropriate runtime explanations is an active research area. Explanations may have to be extensively tailored to the specific goals of the human who needs the explanation (Leake 1991). For example, end-users may not understand an expert description of the competency of the system (Paris 1989). Task architectures may have to be significantly augmented to fully support explanation. Leake argues convincingly that a cache of prior explanations and an active user model are essential components of a good explanation module (Leake 1993). User models and case libraries are not issues addressed in current task formalisms.

2.3.8 We have argued previously that task analysis is falling into the same trap as the conventional knowledge engineering techniques it was evolved to avoid (Compton, Kang et al. 1993). We view knowledge as a dynamic structure that is created in the context of its need. Meta-level inference patterns are still knowledge, therefore these dynamic construct that may not be relevant outside of the domain in which they were primarily evolved .

3. RIPPLE-DOWN-RULES (RDR)

This section describes ripple-down-rules, a system built using test analysis (Compton, Edwards et al. 1991; Compton, Edwards et al. 1992; Kang and Compton 1992; Compton, Kang et al. 1993; Kang and Compton 1993; Mulholland, Preston et al. 1993; Preston, Edwards et al. 1993) .

---

7 For more on tractable abductive inference, see (Bylander, Allemang et al. 1991; Eshghi 1993). For more on generalising abductive architectures, see (Falkenhainer 1990; Poole 1990).

8 For example, after one KADS training course (conducted by specially-imported consultants) for an experienced Australian knowledge engineering group, that group concluded that the overheads of KADS made it unsuitable for projects less than 6 months long (i.e. the majority of their applications).

3.1. DESIGN

In RDR, cases comprising lists of attribute-value pairs are passed one at a time to a propositional rule-base. In classical RDR, each rule can conclude a single classification. The RDR scheme adopts *the frozen knowledge principle* for propositional systems. We "freeze" knowledge that has proved satisfactory. An RDR knowledge base is write-once-only. New knowledge is always an addition to the KB, never a re-write. This is an application of the heuristic: "if it ain't broke, don't fix it".

In RDR, whenever a case results in an inappropriate conclusion, the patch knowledge is entered in as an *unless* test beneath the rule that resulted in the anomaly. As the knowledge base develops, it grows into a binary tree with knowledge patches stored at every node (see figure 1). At runtime, the final conclusion is the conclusion of the last satisfied node.

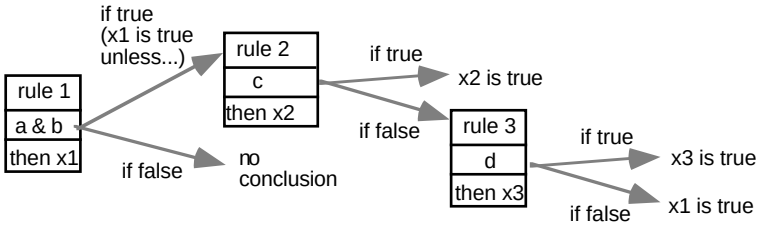

Figure 1: *An RDR-tree. At runtime, the output conclusion is the conclusion of the last satisfied node.*

Note that the RDR formalism makes no commitment to tree structures that are optimal. An RDR tree can contain repeated tests, redundant knowledge, and its sub-trees can overlap each other semantically.

3.2. USE OF TEST CASES

Test cases are used to check for the need to extend the knowledge base. Experts review the output of the system and when they disagree with the system's conclusions, they enter a maintenance environment where they add some *unless* logic beneath the incorrectly last-satisfied node. Only the *logic delta* is added in the new node since the system can not get to this node without first satisfying the logic from the root to the node. So, if *x1* is the correct conclusion when *a & b* is true, but incorrect when *a & b* is true and c is true, then we add the logic delta *c* to a new node on the *if true* branch beneath *rule 1* (as seen in figure 1).

Whenever a new rule is added, the input case that prompted the rule addition is cached with the new rule. Such cases are called *cornerstone cases.* New rules are added by the expert via a simple selection from a *difference list.* To generate this difference list, the RDR shell computes the set difference between all the possible descriptors for the input case and the cornerstone case of the incorrectly last-satisfied node. This is presented to the expert who can select N (N >= 1) items off this list for inclusion into the new rule. For example:

- If the relevant cornerstone case had referred to a measurement of thyroid stimulating hormone (TSH) as "high" and the T4 hormone as "low"; and
- If the incorrectly classified case referred to TSH as "high" and T4 as "high";
- Then the difference list would comprise one item: "T4 = low". References to TSH would be dropped since the TSH attribute(s) of the incorrectly classified case was the same as the TSH attribute(s) of the relevant cornerstone case.

With this difference list, the condition of the patch rule could only be "if T4 = low then ..." .

3.3. TRACES

Explanations within the system are restricted to lists of the fired rules (i.e. a trace of the RDR tree from the route to the incorrectly last-satisfied node), the cornerstone cases found during the inferencing and the difference lists.

3.4. TEST DOMAIN(S)

RDR has been applied to biochemical interpretation and configuration of ion-chromatography detectors (Mulholland, Preston et al. 1993). In one test domain, RDR simulates the reasoning of a doctor studying time series data (the data being a record of the patient's last 5 blood tests) (Compton, Edwards et al. 1992).

3.5. OF COURSE IT WON'T WORK

In this section, we play devils advocate and demonstrate the implausibility of ripple-down-rules.

Representational problems: The models supported by the RDR representation are too simplistic for practical purposes. Consider the nature of the conclusions that can be reached: simple flat facts. Worse still, the system makes no use of intermediary conclusions. Rules cannot make an assertion that another rule uses later in the inferencing. In such a system, how can a knowledge base simulate the progression of a disease?

Feature extraction problems: The system is crucially dependant on a knowledge structure that is represented outside of the system. The list of all possible descriptors for a case, and how we calculate those descriptors from the raw data, is a black box pre-processor. It is not an RDR tree and hence unmaintainable by this system.

Difference list problems: In many real-world domains, the number of descriptions for data (and, in particular, time course data) will be very large. Therefore, the difference lists will be too large and unmanageable.

Domain problems: The language used in the rule conditions is woefully inadequate for expressing complicated medical concepts since it can only perform simple attribute/value comparisons. Pattern matching across time-course data is a non-trivial task (perhaps better solved by genetic algorithms?). Since this data represent a partial snapshot of the disease process, the conclusions reached have a degree of uncertainty about them. We note the concern that RDR has no features for probabilistic reasoning, non-monotonic inference or accessing its previous conclusions when it runs its next inference cycle.

Uncontrolled tree growth: Obviously, the extension of the KB via the addition of patch knowledge beneath each rule will lead to undisciplined KB development. The inability to globally reorganise and optimise the RDR tree will lead to spaghetti knowledge that will needlessly repeat tests as well as patches that repeatedly override some higher-level override. These convolutions in the logic will have two results: (i) slow runtimes; (ii) a graph of the competency of the system vs time will not be a monotonically increasing curve. A patch that fixes one problem could destroy the semantics of the rest of the KB.

Summary: Clearly, ripple-down-rules cannot work. Higher-level abstractions such as the knowledge-level constructs of task analysis are obviously required.

3.6. OH… IT WORKED

RDR is being used for *PIERS*, an expert system for biochemistry at the St. Vincent's Hospital, Sydney. The current system comprises 2037 rules (at the time of this writing) and is in routine daily use. Maintenance time remains constant (measured in rules added per day: average = 3) and the system is maintained by the domain experts without the need for knowledge engineers (Compton, Edwards et al. 1992). Current level of competency: 95% accuracy. The current system is limited to one-fifth of the biochemical tests performed at St. Vincent's Hospital; i.e. using RDR we have been able to construct a system that in a very real pragmatic sense is expert in 20% of human physiology.

PIERS is a significant result for the RDR approach. Previous attempts at building intelligent software for the *PIERS* domain required sophisticated inferencing procedures and experimental causal modelling techniques (e.g. the *ABEL* system (Patil, Szolovitis et al. 1981)). Further these experimental systems never went into routine production.

Experiments with alternative approaches further demonstrate the utility of the RDR approach. Mansuri compared the competency of an rule-base vs time using two approaches: a machine learning technique (ID3) and RDR. Time was measured in number of cases presented. In each approach, a new case was presented to the system, and the rule-base was re-organised appropriately. The RDR system achieved reasonable competency after a few hundred cases (at which time the ID3 system was still 70% inaccurate). ID3 only caught up to RDR after 6000 cases were presented. Note that the RDR system would have been ready for delivery after only a few hundred cases had been collected (Mansuri, Compton et al. 1991).

The following perceived problems proved not to be an issue in practice:

Representational problems: The level of competency of the system suggests that the simple representations of RDR did not hinder the expression of the expert's logic. Either domain experts do not use complicated reasoning in their own thinking or the RDR patch facility permitted adequate repairs on partially expressed concepts.

Feature extraction problems: Feature extraction is not as complicated as one might think (see above paragraph). Also, some degree of repair of poor feature extractors is possible within the RDR trees. For example, suppose the intended logic is "TSH is high, but only a little high". A feature such as "TSH high when TSH > 7.2" can be fixed by patch logic in the RDR to "if TSH > 7.1".

Difference list problems: While the total number of descriptors is large, the difference list is the subset representing the difference between the relevant cornerstone case and the current case. Such a list is manageable in size if a few interface tricks are used (e.g. grouping together differences relating to similar concepts).

Domain problems: Prior to implementation, a consideration of the domain problems lead to an extension of basic RDR for *PIERS*. Extra functions were added to access the last *N* results. Prior results could be tested using *max, min, average, netch, current,* and *previous* functions. Experts could enter in formulae that combined these operators in arbitrary combinations. Preston (Preston, Edwards et al. 1993) studied how these operators were used in practice (studied performed when *PIERS* was at the 957 rule level). Table 1 shows the relative frequencies of use of the various functions in those rules. In the majority of tests conditions, the logic required only

simple tests. The arbitrary combinations of conditions was used very rarely (2.8% of the tests) (Preston, Edwards et al. 1993). The observed competency of the system, despite the in-frequent use of tools that analyse the time course data, suggests that expert reasoning is possible without requiring sophisticated techniques such as non-monotonic inference or probabilistic reasoning.

Function	curr	min	max	netch	previous	average
Used	1069	192	135	108	21	1
Used / 10.69	100	18	12.6	10.1	1.96	0.09

Table 1: *Frequency of functions in 957* PIERS *rules. Line 2 shows line 1 as ratios (1069 = 100 therefore 192 = 18). Taken from* (Preston, Edwards et al. 1993). *Note that while the experts had access to more intricate methods to express their knowledge, satisfactory performance was obtained using very simple constructs.*

Uncontrolled tree growth: RDR trees tend to be broad and flat (maximum number of patches in *PIERS* = 8, average = 2-3). As to the redundancy and overlaps in the logic of the tree, while this is less than optimal in a computational sense, it is somewhat misguided to attempt to optimise an RDR tree to (e.g.) remove the redundancies or separate out the overlaps. The important feature of an RDR tree is that it is optimised for maintenance. Alternative knowledge representation schemes *may* run faster but incur the penalty of harder maintenance. It should be noted that even the seemingly inefficient RDR trees have never proven to be too slow in practice. Further, there exists some experimental evidence that suggests that the redundancy rate may not be significantly large. Gaines and Compton describe techniques for the machine learning of RDR trees. When given cornerstone cases from existing RDR trees, only a 50% size reduction was observed (Gaines and Compton 1992).

3.7. MAINTENANCE SUPPORT

The frozen knowledge principal of RDR simplifies maintenance. To see this, consider how most expert systems would encode the knowledge in the above RDR tree. Most probably, they would enumerate all the logical paths in the RDR tree and write one rule for each path. Assuming the RDR interpreter, then the logical paths for figure 1 are shown in table 2.

Rule	If	Then
1	a & b & not c & not d	x1
2	a & b & c	x2
3	a & b & not c & d	x3

Table 2: *A propositional system that is equivalent to figure 1.*

If the knowledge of the system is patched, then in a conventional rule-based expert system, this patch could extend over many rules. Repeating our above example, the patch on the *x1* anomaly requires an edit to one rule (*rule1*) and the creation of another (*rule2*). Further, the new logic refers to *c* which is a new concept that must be propagated down to all related rules (*rule3*). The more related rules, the more edits. As the knowledge base grows, so to does the number of edits. Hence the time taken to make a change increases and we have a bad maintenance environment.

In RDR, existing knowledge is frozen and we only *extend* the knowledge base. The time taken to change the knowledge does not increase as the knowledge grows since the knowledge base author

does not have to tour all the knowledge to make a patch. Instead, at patch time, the system presents the author with a list of candidate delta logic and the author selects item(s) off that list. These items are added beneath the incorrect node. This is the action at *every* knowledge patch time. Maintenance time is hence reduced.

The astute reader will note that this is only a patch on the rule that was discovered to be anomalous for this case. The same logic anomaly could exist in other rules in different branches and would remain unpatched by the above process. In practice, the process of tracking down these other anomalies occurs as part of correcting other cases[9]. Hence, tracking down anomalies on other branches does not noticeably increase the maintenance effort described above.

4. HYPOTHESIS TESTING

This section describes hypothesis testing: a framework for exploring scientific theories using test analysis (Feldman, Compton et al. 1989; Feldman, Compton et al. 1989; Mahidadia, Compton et al. 1992; Mahidadia, Sammut et al. 1992; Menzies, Mahidadia et al. 1992; Menzies, Compton et al. 1992; Menzies 1993; Menzies and Compton 1993) .

Despite the advantages of the RDR approach (e.g. easy maintenance), RDR makes two assumptions that are not relevant in all domains.

- The expert does not wish to browse the knowledge base. RDR allows experts only to view the portion of the KB used by a particular case. Any other, more global, analysis of the KB is prevented by the RDR interface.
- The KBs generated via RDR will not be used for purposes other than interpretation by an RDR system. A KB generated by a RDR system cannot (e.g.) by ported to a qualitative reasoning system for simulation purposes.

Hypothesis testing is a test analysis technique developed for domains where experts want to browse their hypotheses as well as permitting a migration of the hypotheses for other purposes. The challenge in these domains is to support KB development and maintenance without having to impose on the expert RDR-style restrictions on how they access the knowledge. Further, RDR imposes a KA regime from the very start of the KA process. Hypothesis testing is an experiment in test analysis where KA has already begun and background hypotheses already exist.

4.1. DESIGN

A hypothetical model to be tested *M* is a directed, possibly cyclic and-or graph whose vertices are entities are one of *T* types (where each type can be in one of *S* finite states) and whose edges represent causal effects between entities. One special kind of *T* is a "and" vertex which combines upstream influences to result in a single downstream influence. All other *T* vertices are "or" vertices: arriving at that node from any of its upstream influences gives us permission to propagate an influence downstream. A *useful model* as one that can replicate the observed *behaviour* of the thing that is being modelled. Behaviour is a pair $<C,E>$ where *C* is a set of input causes that result in *E*, the set of output effects.

---

9 As evidence of this, the *PIERS* new rule rate remains constant at 3 rules/day.

Elements of *C* and *E* are state assignments; i.e. associations of a vertex and a state. To generate *E* from *C* and *M*, we search for state assignments of neighbouring vertices that link effects back to any cause. Such an assignment is called an *explanation*. The explanation is given a causal interpretation: vertex V_m in state S_n *caused* adjacent downstream vertex V_o to be in state S_p We attempt to generate one explanation for each element of *E*.

These explanations must not violate a set of invariants *I*. We define invariants in the negative sense; if *not(I)* is true, then no invariants have been violated. Usually, *I* is simply the single rule I_1 which is true (i.e. violated) if a vertex is in two different states. For single explanations, I_1 implies that explanations contain no loops. For the set of explanations required to explain all of *E*, I_1 implies that for a given <*M,C,E*>, explanations require a single state assignment to all vertices used in the explanations.

In our test domain (neuroendocrinology) I_1 often violated: one explanation may require a state assignment that makes another explanation impossible. An *assumption* is a state assignment which commits us to one explanation *or* another (but not *and*). Depending on which assumptions we make, the explanations are divided up into multiple alternative *worlds* W_x, W_y, etc. Within a single world, the explanations do not violate invariants. If a world is a proper subset of another world, and both use the same assumptions, then we fuse them (this condition ensures that we do not deal with the trivial case of one world for each explanation).

A world can be uniquely defined by its *base assumptions* i.e. an assumption which has no upstream assumption. When we move from world to world, we only need to reset the base assumptions, since all other assumptions in that world are dependant on the base assumptions.

The done set *D* is the subset of *E* that can be explained within a world. The *cover* of a world is the cardinality of *D* (i.e. the number of effects that can be explained). We define a "faulty model" as one in which no world W_x can be computed that satisfies:

$$cover(W_x) = |E|$$

4.2. USE OF TEST CASES

For example, consider the mode of figure 2 and the case where:

C = {a(up), b(up)}

E = {d(up), e(up), f(down)}

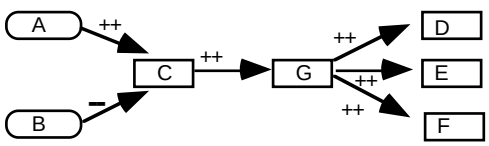

Figure 2A*: Causal connections between the vertices labelled* {a,b,c,d,e,f,g}.

```
cause(up,   '++',  up  ).
cause(down, '++',  down).
cause(up,   '--',  down).
cause(down, '--',  up).
```

Figure 2B: *Definition of edges in figure 2A. For example,* g(up) *could be explained via* c(up) *since the edge* CG *is* '++'.

Note that we have no value for *c* or *g*. Values for *c* and *g* will be assumed as a side-effect of trying to explain *E* in terms of *C*. These explanations are:

E_1 *= {a(up), c(up), g(up), d(up) }*

E_2 = {a(up), c(up), g(up), e(up) }
E_3 = {b(up), c(down), g(down), (down) }

Note that I_1 is violated by *c(up) & c(down)*. We must split our three explanations into two worlds.

W_1 = { E_1, E_2 }
W_2 = { E_3 }.

W_1 contains the assumptions *{c(up), g(up)}* and W_2 contains the assumptions *{c(down), g(down)}*. However, since *g* is fully dependant on *c*, we can define W_1 in terms of the base assumptions *c(up)* and W_2 in terms of the base assumption *c(down)*.

Cover(W_1) = 2 and *cover*(W_2) = 1. Since neither of these covers equals $|E|$; the model is faulty.

Formally, we can characterise the above process as a search for all M_1 and D such that:

[equation 1] $$D \subseteq E \wedge M_1 \subseteq M \wedge (M_1 \wedge D \wedge C \wedge \neg I)$$

4.4. TRACES

Traces of the performance of the causal models are limited to descriptions of how node states influence other node states across the causal links. For typical networks with multiple inputs and multiple outputs, this trace is overwhelming. Graphical depictions of the net cover many screens. These traces have to be summarised to be useful for KA. A list of states between a single output and input look like a simple rule: e.g. *b* going up made *c* go down which made *g* go down which made *f* go down. Domain experts can critique these simple path traces.

4.5. TEST DOMAIN(S)

Hypothesis testing is a far more experimental technique that RDR. Until recently, complexity issues (see below) limited our experimentation with this approach. However, some progress has been made in the processing of time-independent qualitative models as well as causal models. Feldman *et. al.* took quantitative compartmental models[10] of neuroendocrinology and converted them into a qualitative form (Feldman, Compton et al. 1989). Menzies *et. al.* found a more general causal representation underlying the Feldman models that used simpler primitives (Menzies, Mahidadia et al. 1992).

Recently we have come to view these causal models as simpler and/or graph (e.g. figure 2a) (Menzies 1993). By compiling our representations into and-or graphs of possible state transitions, we should be able to apply hypothesis testing to a variety of representations (e.g. mathematical equations, propositional rule-bases (Menzies, Mahidadia et al. 1992)). More speculatively, we have argued previously that hypothesis testing could be the basis for a generic KA loop for multiple model types (Menzies, Mahidadia et al. 1992).

4.6. OF COURSE IT WON'T WORK

In this section, we play devils advocate and demonstrate the implausibility of this test analysis technique.

---

10 Compartmental models utilise the principal of conservation of mass. The sum of flows of substance in and out of a compartment must equal zero. These flows are typically modelled using a time-dependant expodiential function since the rate of flow is often proportional to the amount of stuff in the compartment. For a brief tutorial on compartmental models, see (Menzies and Compton 1993). For more details, see (McIntosh and McIntosh 1980).

Too many behaviours: Given the under-determined nature of the qualitative models processed by hypothesis testing and the feedback loops they contain, surely paths can be found to explain any effect. It would be extremely unlikely that a data set could not be explained in terms of a qualitative model with feedback loops. We note that much of the work in neuroendocrinological theory review has been based on quantitative compartmental modelling. Large databases already exist of quantitative models and experimental results. Why surrender these quantitative models (which support precise prediction) to qualitative models (which can only support the vaguest prediction)?

Better techniques: Better techniques exist for deducing appropriate causal models from data. Pearl describes techniques for deducing causal models (which he terms "belief networks") from observed frequency distributions of variables (Pearl and Verma 1991). His techniques, grounded in sound mathematical theory, are a more principled approach to KA of causal models than the ad-hocary of hypothesis testing.

Fundamental complexity limitations: Given the size of real models, we find it implausible that equation one will be of any practical utility. If consistency is determined via a depth-first chronological backtracking search (e.g. as proposed in (Menzies, Mahidadia et al. 1992)) then the complexity of the process would be overwhelming. Equation one would have to be applied to every subset of the effects and the models; i.e. $O(2^N)$ where $N = |M| + |E|$[11]. We understand that (Menzies 1993) estimates that the search space for a chronological backtracking search for the Feldman '89 model to be 10^{27} (an area much to large to be exhaustively searched).

One technique for taming this complexity would be to impose a meta-structure on the basic node representation. Such a meta-structure would allow us to constrain the search to (e.g.) entities in the same abstract hierarchy. Other research in causal models of human physiology have made extensive use of such a meta-level structure. *ABEL*, developed in the domain of diagnosing acid-base and electrolytic disorders, propagates causal influences over a causal model that exists in N-layers of abstraction (Patil, Szolovitis et al. 1981). The reasoning can move across a level (projecting an hypothesis sideways to find related nodes), up a level (aggregating symptoms to a higher level of abstraction), or down a level (disaggregation: the opposite to aggregation).

Ignoring these meta-level structures implies that search procedures cannot take advantage of them. Note that the Feldman '89 model was constructed from one review article that is a summary of a mere 6 research papers. These 6 papers generated a search space of 10^{27}. Clearly this technique will not scale up.

Better representations: As with our critique of the RDR approach, the representations used here seem too low-level to capture the domain. It is illuminating to consider how other researchers have approached this area. For example, Darden used a theory anomaly detector based on a generic task approach (Darden 1990). Entities within the domain were bundled into groups (using functional knowledge) and anomaly localisation proceeded via this meta-level grouping, rather than at the mere single node level (Moberg 1992). Anomaly localisation was a process of walking backwards from the final state back towards the initial state, through the difference parts of the

---

11 For the model of figure 2A, $|M|$ = the number of arcs = 6 and $|E| = 3$. $2^{6+3} = 512$; i.e. even trivially small models have non-trivial search spaces.

theory (modelled as "function frames") inquiring at each point whether the intermediate state had been entered.

Summary: Clearly, hypothesis testing won't work. Higher-level abstractions such as the knowledge-level constructs of task analysis are obviously required.

4.7. OH… IT WORKED

The success of the hypothesis testing technique was first reported to this Banff workshop in 1989 and 1992. Data sets do exist in the neuroendocrinological literature that cannot be explained via our indeterminate qualitative models.

- Menzies *et. al.* reported anomalies in the *Smythe '87* model (Menzies, Mahidadia et al. 1992). This was a novel finding that Smythe himself was unaware of (Smythe 1992).
- Feldman *et al.* reported that in analysis of the Smythe '89 (Smythe 1989) review paper and 343 data points collected from six related research papers, 109 of the data points could not be explained (32%) (Feldman, Compton et al. 1989). While some of the inconsistencies were due to deliberate simplifications of the model by the researcher, the most important result was that the norepinephrine data in hypothyroid rats who had been given an alpha-2 adrenegeric blocker could not be explained. This was a novel finding that the authors of Smythe '82 research paper (Smythe, Duncam et al. 1982)were not aware of.

That is, we can fault published neuroendocrinological theories using previously-unrecognised relationships in the data published to support them. Further, the anomalies that we found were previously undetected. We offer two theories for why this is so:

1) *Focus:* Researchers design experiments to test a particular hypothesis. Our fault detection often occurs in comparisons of data that was not relevant to the hypothesis they set out to test.

2) *Mental effort:* Without an automatic tool to perform the search for causal pathways, even simple models are too complex to process by hand. In larger models, the complexity of the task is truly overwhelming (recall the 10^{27} figure mentioned above).

As regards the other objections raised above:

Fundamental complexity limitations: Our two initial prototypes were, in retrospect, naive in their search and would not have scaled up. The Feldman '89 study could only process 28 of the 992 experimental comparisons (i.e. those with only one cause and no steady effects) and took two days to run. The work on hypothesis testing over the last year has focused on the complexity problem and we have now moved beyond the chronological backtracking approach used in (Menzies, Mahidadia et al. 1992) to an assumption-based truth-maintenance (ATMS) approach that builds all alternative world simultaneously (thus avoiding the complexity of re-inventing each alternative world as the search tries each alternative). When analysing different worlds, we now only iterate over the base assumptions (rather than over all assumptions). Such base assumptions are calculated as a side-effect of applying our ATMS algorithm. The new approach can process all 992 comparisons in under 5 hours using an interpreted language (Smalltalk). Moving to a faster language (e.g. "C") could speed that time up by a factor of up to 40. Also, recent

observations of the behaviour of the search suggest that further reductions in the complexity may be possible. (Menzies 1993).

Better techniques: Belief networks do not permit the integration of an existing theory by an analysis of a new example. In our system, we can foresee that various users would treasure their favourite portions of the model (typically, the ones they have developed and successfully defended from all critics). It would be unacceptable to permit an algorithm to scribble all over this knowledge.

Also, generating belief networks requires access to large amounts of data is available on all the entities in the domain. Our domain is characterised as being hypothesis-rich, but data-poor. Obtaining values for certain chemicals within the body is not as simple as, say, attaching a volt meter to an electric circuit. Often delicate measurements have to be made by skilled staff using expensive equipment. The measurement must be repeated a statistically meaningful number of times. Also, in certain domains, it may take years to gather the data (e.g. large-scale epidemiological studies). Hence, the data required to assist feuding experts debating different versions of the same model may be unobtainable or incomplete. Consequently, in the Feldman '89 study, over 90% of the nodes were unmeasured.

Better representations: We found that the meta-level constructs of *ABEL* and the Darden/Moberg study were not required for fault detection. *ABEL*'s causal links are more complicated than in hypothesis testing: they stored information about causal severity and duration. Patil *et. al.* do not justify the complexity of their system. The essential feature of the Darden/Moberg representation that supported model faulting was the causal links between entities in the domain. In terms of model faulting, the rest of the architecture was superfluous. In fairness to their work we should add that, unlike our work, they were exploring an existing representation rather than seeking the minimal architecture needed for model refutation. At most, we could argue that the Darden/Moberg study demonstrated that in terms of model revision, the useful features of a representation are the causal links between entities. At the very least we observe that verification does not fall out straight away from a task analysis approach, but requires some additional work.

4.8. MAINTENANCE SUPPORT

The qualitative nature of hypothesis testing, and its ability to handle missing data, makes our approach more suitable as a review/maintenance tool for scientific models than alternative techniques (e.g. qualitative compartmental modelling). The models and data we processed were taken from publications; i.e. our detected anomalies had escaped peer review and the inspection of the international neuroendocrine community.

The drawback with quantitative compartmental modelling is that it requires extensive experimental observations. Much of the time of the neuroendocrinologists is spent collecting data on a parameter of a proposed link in a compartmental model. A single qualitative causal reference translated into a quantitative equation may require numerous values before that equation can be executed. If the macro-structure of the proposed model is wrong, then this effort is wasted. Hypothesis testing allows a researcher to quickly sketch and test the macro-structure of a model before wasting time on collecting data on potentially spurious connections.

5. DEBATE

In this section, we try to anticipate our critics and discuss objections to test analysis.

5.1. THE NEED FOR EXPLANATION

Objection: Expert systems need to support knowledge-level explanations. Such explanations can be generated via traces of execution of task architectures. Therefore, we should use task analysis to develop expert systems, particularly when we are building tutoring systems.

Let us distinguish between two uses of explanation: (1) explanations for KA purposes and (2) explanations requested by users at runtime from the delivered system:

Explanation for KA: We agree that expert systems need to supply "clues" regarding how the inference traverses the knowledge base in order to detect knowledge anomalies. For performance systems, these clues need not be knowledge-level descriptions, providing that they support KB anomaly detection and repair. RDR and hypothesis testing are examples of systems where the clues are below the knowledge level, yet competent expert systems can be developed.

Explanation for end-users at runtime: As to runtime explanations for end-users from the delivered expert system, our own experience has been that (a) this is a little-used feature and (b) satisfying justifications can be supplied without recourse to knowledge level descriptions. Recall that *PIERS* has no explanation other than the rule trace. No doctor, other than the rule development team, has ever requested an explanation from *PIERS*. When the development team wanted explanations, the rule trace and a list of cornerstone cases along the rule-trace sufficed. Menzies' *PIGE* system was an intelligent post-processor to a mathematical model of a pig growing in a piggery. As well as the recommendation screens of the expert system, the users could access detailed mathematical analyses that explained the system's reasoning. The experience there was that as long as the end-users knew that they could access the mathematical reports, that they never actually did so. Menzies *et al.* conjecture that expert systems don't need an explanation facility at runtime, but an excuse facility that gave users a feeling that they could trust the reasoning (Menzies, Black et al. 1992).

Lastly, as noted in §2.3.7, the problem of explanation is not solved merely by offering a trace of the system’s traversal over a task description. Explanation is an active research area and general principles are only just beginning to develop (Paris 1989; Leake 1991; Leake 1993).

5.2. LACK OF GENERALITY

Objection: Test analysis is hardly a general technique while task analysis is relevant to more domains.

While it is true that RDR has only been successfully applied to well for classification tasks, attempts to generalise the technique to other domains have been encouraging (but only partial successful). Mulholland's *et al.* ion chromatography system was an experiment in extending RDR into configuration tasks. Gaine's *INDUCT* machine learning program was used to generate multiple RDR trees from libraries of known chromatography configurations (Gaines and Compton 1992). Each tree related to the setting of a particular parameter. A control structure was then placed over the trees such that they were called in an appropriate sequence for the

configuration. KA sessions could patch the RDR trees using the RDR interface, but the control structure was outside of the RDR maintenance environment (Mulholland, Preston et al. 1993).

However, we believe it possible that, with further research, test analysis can be become a general technique. Compton argues that RDR can be used to initialise a system which can then incrementally grow into a classification or configuration or whatever type of system (Compton, Kang et al. 1993). Our preliminary work with hypothesis testing suggests that it can be applied to qualitative reasoning, causal modelling, and propositional rule-bases (Menzies, Mahidadia et al. 1992). Further, our techniques seem very similar to the general notion of expert critiquing systems (ECS) (Silverman 1992) which Silverman defines as follows:

> *"programs that first cause their user to maximise the falsifiability of their statements and then proceed to check to see if errors exist. A good critic program doubts and traps its user into revealing his or her errors. It then attempts to help the user make the necessary repairs*[12]*".*

Silverman's research seems to be focused on an implementation-independent analysis of the process of "critiquing" a program. Hypothesis testing aims for an engineering formalism that allows us to simultaneously process and validate different types of knowledge.

5.3. INCORRECT MODELS

Objection: The validity of a model can be measured by more than just its ability to cover certain test cases. Other measures such as parsimony, succinctness, and clear use of existing domain concepts are just as important as performance. A narrow focus merely on performance could give rise to incorrect and truly bizarre models indeed.

We have two comments on this objection. Firstly, we note that tacit in this objection is a belief in the now-out-dated knowledge-transfer approach. Gone are the days when KA was viewed as the process of extracting the experts actual knowledge (poetically described by Feigenbaum as "mining the jewels in the expert's head" (Feigenbaum and McCorduck 1983)). Note that *none* of the above points 2.1.1 to 2.1.7 makes any claim to the meta-level patterns being an instanitation of an actual model of human cognition[13]. Davis *et al.* give the modern view of knowledge representation: representations are an inaccurate surrogate of reality (Davis, Shrobe et al. 1993). Gaines report that "the *knowledge-modelling* perspective has become widely adopted and terminologies reflecting an expertise-transfer perspective have been quietly dropped" (Gaines 1992).

We endorse the knowledge modelling perspective and believe that it is folly to aim for the "right" or "correct" knowledge base for an expert system. Expert systems are models of the world and since models are not the same as the thing they are modelling, *they can never be completely correct*. Popper argues that our models can never be "proved" in some absolute sense. Instead, the currently "true" theories are merely the ones that have not been disproved (Popper 1963). We have argued previously that knowledge is a dynamic construct and that our KBs should be tools for exploring those constructs rather than tools for enshrining old insights (Compton and Jansen

---

12 ECSs are therefore much broader than the definition instantiated by *ATTENDING* (Miller 1986) which had no mechanism for doubting its own internal knowledge base.

13 Though (O'Hara and Shadbolt 1993) notes that despite their official public line, some task analysis researchers believe in a human psychological basis for tasks.

1990). KA is a process of building and debugging a symbolic model that may never have been created previously. This view is endorsed by Bradshaw et al who argue that the primary utility of a model is:

> *…not to serve as a "picture" of a domain, but as a device for the attainment or formulation of knowledge about it. (Bradshaw, Ford et al. 1991)*

We regard the (i) RDR trees with their redundancies and overlapping subtrees or (ii) the neuroendocrinological causal models with redundant or unproven links as being just as valid as any other construct for these domains. We demand only that our KBs exhibit competency in their domain and can provide clues regarding knowledge anomalies.

Secondly, "clear use of existing domain concepts" is an interesting issue. Medicine, for example, has entire libraries devoted to expounding domain concepts. Note that medical expert systems use very few of these concepts and often have to invent new constructs (e.g. the context tree in MYCIN (Buchanan and Shortliffe 1984)). *PIERS* (which can process 20% of all biochemical tests at St. Vincent's Hospital) used *none* of these concepts. Clancey's research on generating explanations from MYCIN showed that the knowledge required to explain an drug recommendation was different to the knowledge require to make the recommendation (Clancey 1983; Clancey 1984). We would argue that many "existing domain concepts" have been evolved for explanation purposes and may not be relevant for competency.

6. CONCLUSION

We have described our progress towards a general test analysis methodology (summarised in table 3). In domains where the experts do not wish global knowledge of the KR, RDR can be useful. In domains where global browsing is required, hypothesis testing may be appropriate. We concede that test analysis requires more work but claim that our current results are encouraging.

Technique	**Domain**	**"Naive" KA approach**	**Pre-experimental intuition**	**Experimental result**	**Augment or replace task-based KA?**
Ripple-Down-Rules	Classification	Binary tree with logic patches	Tree will be unmanageable. Competency unlikely.	Large systems (2037 rules) can be built and easily maintained without knowledge engineers.	Replace.
Hypothesis testing	Qualitative models of the neuro-endocrine system	Indeterminate causal influences.	Indeterminacy will allow any behaviour to be generated. Ability to critique hypotheses unlikely.	Naturally occurring data sets can be faulted (these faults are invisible to other techniques)	Possibility of replace.

Table 3: *Summary of test analysis techniques. Competent performance systems can be built using simplistic techniques. Note that the pre-experimental intuitions regarding the naivety of the techniques were wrong. Perhaps before moving on to more complicated techniques, we should experiment further with simpler alternatives.*

Issues relating to testing are often deferred during discussions about KA and KR design. An attitude we commonly strike is "oh yes, we can test it ... later". Here we have argued for (i) test sooner rather than later and (ii) that development methodologies based around rigourous testing

can replace alternative development techniques. The constructs used in a KR built using testing analysis were significantly simpler than constructs required by other approaches. We wonder if underlying the intricacies of existing expert systems are a minimal set of KR techniques that are the essential components of artificial expert competency. We ask proponents of alternative methodologies requiring more intricate design constructs two questions:

- Have you experimented with simpler alternatives? We note that designs that seem naive at first glance may in fact produce satisfactory competency with comparatively less effort.
- Does your approach support the maintenance cycle?

So, when can "test" replace "task"? We have given examples of performance systems that were developed using test analysis. Such systems are unsuitable for the following purposes: (i) a teaching tool, or (ii) a theoretical tool for generalising old designs. We note that once we have developed a tested performance system, then task analysis can be used reverse-engineering a knowledge-level description of that system. That is, for performance systems, testing replaced task but task analysis could augment test analysis once a system was in production. Indeed, this reverse engineering for a succinct description is what task analysis was designed for and may be its most useful feature.

7. REFERENCES

Aben, M. (1992). *On the Specification of Knowledge Model Components*. **Formal Methods for Knowledge Modelling in the CommonKADS Methodology: A Compilation (KADS-EE/T1.2/TR/ECN/014/1.0).** Netherlands Energy Research Foundation. 110-127.

Akkermans, H., van Harmelen, F., Schreiber, G. and Wielinga, B. (1992). *A Formalisation of Knowledge-Level Models for Knowledge Acquisition*. **Formal Methods for Knowledge Modelling in the CommonKADS Methodology: A Compilation (KADS-EE/T1.2/TR/ECN/014/1.0).** Netherlands Energy Research Foundation. 53-90.

Allemang, D. (1992). **Modelling a configuration problem with generic tasks**. GMD (Gesellschaft fur Mathematik und Datenverarbeitung mbH).

Bradshaw, J. M., Ford, K. M. and Adams-Webber, J. (1991). *Knowledge Representation of Knowledge Acquisition: A Three-Schemata Approach*. **6th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop, ,October 6-11 1991**, Banff, Canada,

Buchanan, B. G. and Shortliffe, E. H. (1984). **Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project**. Addison-Wesley.

Buchanan, B. G. and Smith, R. G. (1989). *Fundamentals of Expert Systems*. **The Handbook of Artificial Intelligence, Volume 4.** Addison-Wesley. 149-192.

Bylander, T., Allemang, D., Tanner, M. C. and Josephson, J. R. (1991). *The complexity of abduction.* **Artificial Intelligence** 49: 25-60.

Chandrasekaran, B. (1983). *Towards a taxonomy of problem solving types.* **AI Magazine** (Winter/Spring): 9-17.

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

Chandrasekaran, B. (1990). *Design Problem Solving: A Task Analysis.* **AI Magazine** (Winter): 59-71.

Clancey, W. (1983). *The epistemology of rule-based systems: a framework for explanation.* **Artificial Intelligence** 27: 289-350.

Clancey, W. (1984). *Use of MYCIN's rules for Tutoring.* **Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project.** Addison-Wesley. 464-489.

Clancey, W. (1985). *Heuristic Classification.* **Artificial Intelligence** 27: 289-350.

Clancey, W. J. (1992). *Model Construction Operators.* **Artificial Intelligence** 53: 1-115.

Compton, P., Edwards, G., Kang, B., Lazarus, L., Malor, R., Menzies, T., Preston, P., Srinivasan, A. and Sammut, C. (1991). *Ripple down rules: possibilities and limitations.* **6th Banff AAAI Knowledge Acquisition for Knowledge Based Systems**, Banff, Canada,

Compton, P., Edwards, G., Srinivasan, A., Malor, P., Preston, P., Kang, B. and Lazarus, L. (1992). *Ripple-down-rules: Turning Knowledge Acquisition into Knowledge Maintenance.* **Artificial Intelligence in Medicine** 4: 47-59.

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

Compton, P. J. and Jansen, R. (1990). *A philosophical basis for knowledge acquisition.* **Knowledge Acquisition** 2: 241-257.

Darden, L. (1990). *Diagnosing and Fixing Faults in Theories*. **Computational Models of Scientific Discovery and Theory Formation.** San Mateo, California., Morgan Kaufmann Publishers Inc.

Davis, R., Shrobe, H. and Szolovits, P. (1993). *What is a Knowledge Representation?* **AI Magazine** (Spring): 17-33.

Eshghi, K. (1993). *A Tractable Class of Abductive Problems*. **IJCAI '93**, Chambery, France,

Falkenhainer, B. (1990). *Abduction as Similarity-Driven Explanation*. **Working Notes of the 1990 Spring Symposium on Automated Abduction.**, UC Irvine.

Feigenbaum, E. and McCorduck, P. (1983). **The Fifth Generation**. New York, Addison-Wesley.

Feldman, B. T., Compton, P. J. and Smythe, G. A. (1989). *Towards Hypothesis Testing: JUSTIN, Prototype System Using Justification in Context*. **Proceedings of the Joint Australian Conference on Artificial Intelligence, AI '89**,

Feldman, B. Z., Compton, P. J. and Smythe, G. A. (1989). *Hypothesis Testing: an Appropriate Task for Knowledge-Based Systems*. **4th AAAI-Sponsored Knowledge Acquisition for Knowledge-based Systems Workshop**, Banff, Canada, October 1989.,

Gaines, B. (1992). **AAAI 1992 Spring Symposium Series Reports: Cognitive Aspects of Knowledge Acquisition**. AI Magazine. 24.

Gaines, B. and Compton, P. (1992). *Induction of Ripple Down Rules*. **AI '92**, Hobart, Tasmania, World Scientific.

Hamscher, W., Console, L. and J., D., Ed. (1992). **Readings in Model-Based Diagnosis**. San Mateo CA, Morgan Kaufmann.
Hewlett-Packard (1993). *DX '93*. **Fourth International Workshop on Principles of Diagnosis**, University College of Wales, Aberyswyth, Wales, United Kingdom,
Kahn, G., Nowlan, S. and McDermott, J. (1985). *Strategies for Knowledge Acquisition.* **IEEE Transactions on Pattern Analysis and Machine Intelligence** Vol. PAMI-7(No. 5. September): 511-522.
Kang, B. and Compton, P. (1992). *Towards a process memory*. **AAAI Spring Symposium: Cognitive Aspects of Knowledge Acquisition**, Stanford University,
Kang, B. and Compton, P. (1993). *Knowledge acquisition in context: the multiple classification problem*. **Pacific Rim International Conference on Artificial Intelligence**, Seoul,
Larkin, J., McDermott, J., Simon, D. P. and Simon, H. A. (1980). *Expert and Novice Performance in Solving Physics Problems.* **Science** 208(20 June): 1335-1342.
Leake, D. B. (1991). *Goal-Based Explanation Evaluation.* **Cognitive Science** 15: 509-545.
Leake, D. B. (1993). *Focusing Construction and Selection of Abductive Hypotheses*. **IJCAI '93**,
Linster, M. and Musen, M. (1992). *Use of KADS to create a conceptual model of the ONCOCIN task*. **Knowledge Acquisition** 4(1): 55-88.
Mahidadia, A. J., Compton, P., Menzies, T. J., Sammut, C. and Smythe, G. A. (1992). *Inventing Causal Qualitative Models: A Tool for Experimental Research*,. **Proceedings of AI '92**, Hobart, Tasmania, November, 1992,
Mahidadia, A. J., Sammut , C. and Compton, P. (1992). *Building and Maintaining Causal Theories*. **AAAI Symposium on Knowledge Assimilation**, Stanford University, Spring, 1992.,
Mansuri, Y., Compton, P. and Sammut, C. (1991). *A comparison of a manual knowledge acquisition method and an inductive learning method*. **Australian workshop on knowledge acquisition for knowledge based systems**, Pokolbin, Australia,
Marques, D., Klinker, G., Dallemangne, G., Gautier, P., McDermott, J. and Tung, D. (1991). *More data on useable and reusable programming constructs*. **6th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop**, Banff, Canada,
McIntosh, J. E. A. and McIntosh, R. P. (1980). **Mathematical Modelling and Computers in Endocrinology**. Berlin, Springer-Verlag.
Menzies, T., Mahidadia, A. and Compton, P. (1992). *Using Causality as a Generic Knowledge Representation, or Why and How Centralised Knowledge Servers Can Use Causality*. **Proceedings of the 7th AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop**, Banff, Canada, October 11-16,,
Menzies, T. J. (1993). *The Complexity of Model Review*. **Dx -93: The International Workshop on Principles on Model-Based Diagnosis**, Absersywath, Wales, UK,
Menzies, T. J., Black, J., Fleming, J. and Dean, M. (1992). *An Expert System for Raising Pigs*. **Th first Conference on Practical Applications of Prolog**, London, UK.,
Menzies, T. J. and Compton, P. (1993). **Whiteboard Physics**. AI Lab, University of New South Wales.
Menzies, T. J., Compton, P. and Mahidadia, A. (1992). *Communicating Research Models of Human Physiology using Qualitative Compartmental Modelling*. **Communicating Scientific and Technical Knowledge, a AAAI '92 workshop**, San Jose, California, July 12-17,
Miller, P. L. (1986). **Expert Critiquing Systems**. Spriner-Verlag.
Moberg, D. (1992). **Personal communication**.
Mulholland, M., Preston, P., Hibbert, B. and Compton, P. (1993). *An expert system for ion chromatography developed using machine learning and knowledge in context*. **Sixth International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems**, Edinburgh,
Newell, A. (1982). *The knowledge level.* **Artificial Intelligence** 18: 87-127.
O'Hara, K. and Shadbolt, N. (1993). *AI Models as a Variety of Psychological Explanation*. **IJCAI**, Chambery, France,
O'Rourke, P. (1990). **Working Notes of the 1990 Spring Symposium on Automated Abduction**. University of California, Irvine, CA.
Paris, C. L. (1989). *The Use of Explicit User Models in a Generation System for Tailoring Answers to the User's Level of Expertise*. **User Models in Dialog Systems.** Springer-Verlag. 200-232.
Patil, R. S., Szolovitis, P. and Schwartz, W. B. (1981). *Causal Understanding of Patient Illness in Medical Diagnosis*. **IJCAI '81**,
Pearl, J. and Verma, T. S. (1991). *A Theory of Inferred Causation in*. **Principles of Knowledge Representation and Reasoning, Proceedings of the Second International Conference**, Morgan Kaufmann.
Poole, D. (1990). *Hypo-Deductive Reasoning for Abduction, Default Reasoning, and Design*. **Working Notes of the 1990 Spring Symposium on Automated Abduction.**, UC Irvine.
Popper, K. R. (1963). **Conjectures and Refutations,**. London, Routledge and Kegan Paul.
Preston, P., Edwards, G. and Compton, P. (1993). *A 1600 rule expert system without knowledge engineers*. **Second World Congress on Expert Systems**, Lisbon, Pergamon.
Schreiber, A. T., Wielinga, B. J. and Akkermans, J. M. (1992). *Using KADS to Analyse Problem Solving Methods*. **Formal Methods for Knowledge Modelling in the CommonKADS Methodology: A Compilation (KADS-EE/T1.2/TR/ECN/014/1.0).** Netherlands Energy Research Foundation. 53-90.
Silverman, B. G. (1992). *Survey of Expert Critiquing Systems: Practical and Theoretical Frontiers.* **Communications of the ACM** 35(4): 106-127.
Smythe, G. A. (1989). *Brain-hypothalmus, Pituitary and the Endocrine Pancreas*. **The Endocrine Pancreas.** New York, Raven Press.
Smythe, G. A. (1992). **Concerning errors in the Smythe '87 model**.
Smythe, G. A., Duncam, M. W., Bradsahw, J. E., Cai, W. Y. and Symons, R. G. (1982). *Control of Growth Hormone Secretion: Hypothalmic Dopamine, Norepinephrine and Serotonin Levels and Metabolism in Three Hyposomatrophic Rat Models and in Normal Rats.* **Endocrinology** 110(2): 376-383.
Steels, L. (1990). *Components of Expertise.* **AI Magazine** 11(2): 29-49.
Stefik, M., Aikins, J., Balzer, R., Benoit, J., Birnhaum, L., Hayes-Roth, F. and Sacerdoti, E. (1982). *The Organisation of Expert Systems, A Tutorial.* **Artificial Intelligence** 18: 135-127.
Tansley, D. S. W. and Hayball, C. C. (1993). **Knowledge-Based Systems Analysis and Design**. Prentice-Hall.
Tu, S., Shahar, Y., Dawes, J., Winkles, J., Puetra, A. and Musen, M. (1991). *A Problem-Solving Model of Episodic Skeletal-Plan Refinement*. **6th BANFF Knowledge Acquisition for Knowledge-Based Systems Workshop**, Canada,
Van de Brug, A., Bachant, J. and McDermott, J. (1986). *The Taming of R1*. **IEEE Expert** (Fall): 33-39.
Wielinga, B. J., Schreiber, A. T. and Breuker, J. A. (1992). *KADS: a modelling approach to knowledge engineering.* **Knowledge Acquisition** 4(1): 1-162.