More Results on the Practical Lower Limits of Test Set Size

Tim Menzies[1], Sam Waugh[2]

[1]Artificial Intelligence Department,
School of Computer Science and Engineering,
University of NSW, Australia, 2052
[2]Defence Science and Technology Organisation,
Air Operations Division, Melbourne, Australia, 3001
tim@menzies.com
sam.waugh@dsto.defence.gov.au
http://www.cse.unsw.edu.au/~timm

August 4, 1998

Abstract

A general experimental rig is described for detecting where validation fails. This rig can offer precise domain-specific recommendations regarding minimum test set size. In the best case found to date, as few as 13 tests were sufficient to demonstrate the competency of a KBS. Further, the size of each test was very small: they ignored ninety percent of the KBS. *Submitted to the Pacific Knowledge Acquisition Workshop, PKAW '98, Singapore, November, 1998.*

1 Introduction

Modern knowledge acquisition views knowledge-based system (KBS) construction as the construction of inaccurate surrogates models of reality [8,40]. Agnew, Ford and Hayes [1] comment that *expert-knowledge is comprised of context-dependent, personally constructed, highly functional but fallible abstractions.* Practioners confirm just how inaccurate systems can be:

- Silverman [35] cautions that systematic biases in expert preferences may result in incorrect/incomplete knowledge bases.
- Compton [6] reports expert systems in which there was always one further important addition, one more significant and essential change.
- Working systems can contain multiple undetected errors. Preece and Shinghal [31] document five fielded expert systems that contain numerous logical anomalies. Myers [27] reports that 51 experienced programmers could only ever find 5 of the 15 errors in a simple 63 line program, even given unlimited time and access to the source code and the executable.

Potentially inaccurate and evolving theories must be validated, lest they generate inappropriate output for certain circumstances. Testing can only demonstrate the presence of bugs (never their absence) and so must be repeated whenever new data is available or a program has changed. That is, testing is an essential, on-going process through-out the lifetime of a knowledge base.

Testing a theory should be impossible. A theory with N variables with S states (on average) may require up to S^N tests. In practice, this number is impossibly large. For example, one sample of fielded expert systems contained between 55 and 510 literals [31]. Literals offer two states for each proposition: true or false (i.e. $S=2$ and N is half the number of literals). Assuming that (i) it takes one minute to consider each test result (which is a gross under-estimate) and (ii) an effective working day of six hours a day and 225 days a year, then a test of those sampled systems would take between 29 years and 10^{70} years (a time longer than the age of this universe).

The above analysis must be somehow naive. Most test regimes are defined using far less than S^N tests:

- Caraca-Valente *et.al.* [3] report that many expert system writers recommended small test sets of between five to fifty items.
- Menzies has described a detailed evaluation of a process control expert system for a petrochemical plant with many thousands of variables. That evaluation included a comparative analysis of the expert system with a human expert. The evaluation required only 40 tests [21].
- Object-oriented analysts and designers recommend specifying an their systems with between 10 to 100 short, specific examples of system operation (the *use case*) [14].
- Usability engineers have explored the ideal number of testers for an interface. This cost-benefit curve plateaus at two to four users: a remarkably small number [29].

If S^N is naive, how many tests would a smart analysis propose? Can we say, practically speaking, what are the minimum number of tests for a system? This is an important practical problem for KBS developers. In many domains, it is difficult and expensive to find or build test sets. For example:

- The (in)famous *Limits to Growth* study attempted to predict the international effects of continued economic growth [17]. Less than 0.1 percent of the data required for the theories was available [5].
- Data collections in neuroendocrinology (the study of nerves and glands) can be just as sparse since data collection in that domain is very expensive. In one extreme example, 300,000 sheeps brains had to be filtered to extract 1.0 milligrams of purified thyroptin-releasing hormone [16].
- Techniques exist for automatically generating test sets. For example, the dependency network of a system can be used to determine inputs that will exercise all branches of the system. Sophisticated non-monotonic techniques can be used to separate inputs into sensible subsets [11, 41]. However, once an input suite is inferred, an expert still has to decide what are the appropriate outputs for those inputs. This may be a significant analysis task and, in practice, may only be practical for small systems.

Given the practical problems associated with data collection, practioners often need to rationalise the process of building test sets. The value of building bigger test sets must be weighed up against the cost of their construction. To avoid wasting money, practioners must build test sets big enough to be useful, but no bigger. This paper explores *how big is big enough?*. Assuming that the goal of each test will to be reach some outputs *out.i* from some inputs *in.i*, then we define test set size and test item size as follows:

- *TEST SET SIZE*: The size of a test set N will be number of pairs: *((in.1,out.1), (in.2,out.2), ..., (in.N,out.N))*.
- *TEST ITEM SIZE*: The size of a single test item will be computed from *(in.i,out.i)* and denoted U: the percentage of theory variables *not* referenced in *in.i* and *out.i*. At U*=0*, no variable is *U*nmeasured. At U*=100*, everything is unmeasured (i.e. *in.i* and *out.i* are empty). As U decreases, more data has to be collected and the cost of the test increases. As U increases the test is cheaper but it is less likely that the test will detect anything.

This article is structured as follows. We begin with some preliminary notes. Based on a literature review, it will be argued that the lower useful bound on test set size is 13 to 30 [3, 4, 7, 37]. Experiments with a general validation engine will show that the practical limits on the size of single test item are quite variable. Prior work identified classes of theories where testing was practical for *U=0..70* [25] (see below). This study reports another class of theories where testing is practical for *U=0..90.*

1.1 Preliminary Notes

The terminology of this paper supersedes previous publications in this area [18–20, 23–25, 39].

While this paper is written in the context of knowledge engineering, there is nothing in principle stopping the application of these results to standard software engineering.

A knowledge base, as viewed by most of this article, is either (i) a black-box with inputs and outputs or (ii) a directed graph connecting knowledge-base literals. This view is orthogonal to the standard view of a knowledge base as domain knowledge plus ontology plus problem solving method (PSMs); e.g. [12,34]. The analysis of this article could be applied to PSMs and ontologies if some partial evaluation system (e.g. [33]) converted them to ground horn clauses (ground horn clauses can be viewed as a directed graphs with sub-goals connected via an AND-node to the head).

This works assumes a model of testing of the following form: *can a theory of X reproduce known or desired behaviour of X?* (where the known or desired behaviour of X is stored in a test set). Other models of testing exist (e.g. the syntactic anomaly detection work of Preece [30]). There are at least two advantages of the testing model used in this paper. Firstly, precise limits to this model of testing can be determined (see below). Limits to test in other models of testing is an open issue and, to the best of our knowledge, not actively researched Secondly, if a system fails our model of testing then (assuming the test set is correct), then something *must* be wrong. The same cannot be said for other testing models. For example, Preece stresses that his work does not detect *errors*; rather it only detect *anomalies* which require further human investigation. That is, if a system fails a Preece-style check, it is still possible that nothing is really wrong.

Machine learning researchers may be surprised at this article's endorsement of small test sets. Much of machine learning research is focused on a summary of large amounts of examples (e.g. [26, 32]). A truism in that field is that *the more data, the better*. However, the goal of this research is the assessment of an existing theory, not the creation of a new theory. Theory creation may require far more data than 13 to 30 examples.

The paradigm of this article is exactly the kind of thinking rejected by Newell [28]. Reacting to an excess of experimental zeal in cognitive psychology, Newell argued forcibly against conducting experimental programmes that offered single answers to yes-no questions. Rather, said Newell, we should perform multiple studies of complex systems, collect large amounts of data, and unify those results into some rich theoretical structure. Such large simulations, he argued, are far more insightful than (e.g.) twenty yes-no questions. Indeed, we will show below cases were far more than twenty questions are be required (e.g. when errors are detected, or when more than one conclusion is required). However, Newell's comments are not fatal to this research. For many applications, we seek not to learn new laws about the universe. Rather, we only seek to certify the competency of a particular device. Our *lower bounds on test sets* argument applies only to the certification problem. Also, Cohen (personal communication, 1998) argues that Newell confused the specifics of an observation with the generality of the implication of that observation. For example, consider the question *do heavy objects fall faster in vacuum than light objects?*. The *yes* answer would have enormous implications for our vision of the universe since it would challenge base assumptions about gravity.

2 Lower Bounds on Size of Test Set

The next section argues that, in the best case, the size of a single test item may be quite small (i.e. it can ignore up to ninety percent of the theory). This section reviews mathematical arguments that the number of tests can be very low. First, an optimistic argument will be presented that the number of tests can be 30 or less. Next, some cautionary remarks are offered to balance this optimism.

More Than 50 is Too Much: Various researchers report that a large test set size can confuse, not clarify. Courtney and Gustafson [7] warn that numerous spurious correlations can be found in large sample sizes. Cohen notes that the standard error on the mean degrades steeply up to around 30 samples, and degrades very slowly after 50 samples [4]. That is, the benefit of performing more than 50 tests is dubious.

20 to 30 Tests are Enough: A *normal distribution* is a well studied curve in statistics. It can be fully characterised by the mean and spread (variance) of this bell-shaped curved. Once a distribution is characterised as being normal, then values can be predicted with well-defined degrees of confidence. Not all distributions are normal. However, the *central limit theorem* shows that if 30 random samples are taken from any distribution, then that sample can be approximated using the normal distribution [37]. While 30 is desirable, this approximation is serviceable after 20 samples. That is, test sets larger than 30 items can be approximated by 30 or less randomly generated test sets.

13 Tests Are Often Enough: Caraca-Valente *et.al.* [3] studied the margin of errors found when testing expert systems for the physically handicapped. Those systems handled a variety of tasks including homeopathic treatment, adapting jobs and vocational guidance, and physiotherapeutic diagnosis. Empirically, they noted an exponential decay relationship between the number of test cases and the maximum error from each test. As the number of tests decreased, the maximum error increased rapidly until the knee of the curve. After the knee, the maximum error grew much slower; that is, after some point, there was less and less benefit in increasing the test set size. Caraca-Valente *et.al.* explored if this was a quirk of their application domain. Like the central limit theorem, they assumed random samples. They offer five results:

1. A theoretical relationship between the number of tests made and the error on the results of those tests. This relationship includes three special parameters.

2. A method of deriving the domain special parameters used in that relationship using a least-squares estimation.

3. Using these first two results, it was shown that their empirical data can be reproduced from their theoretical analysis.

4. A set of sample curves for that relationship.

5. A second relationship showing the theoretical ideal number of tests.

A common feature of results three, four, and five was that for a variety of systems (including their actual expert systems), the knee in all the curves was never more than 13 tests (and sometimes went down as low as five). Caraca-Valente *et.al.* offer guidelines for determining when some application will require more than 13 tests, but offer no example of such systems.

Cautionary Remarks: The above analysis has certain limitations:

- The above results only comment on how many tests are required to check if a system passes some criteria. If a system was to fail that check, then many more tests may be required to diagnosis the fault (for more on fault localisation, see [13]).

- The above tests may have to be repeated for each conclusion required. For example, 40 tests are required for a comparative evaluation of human operators vs an expert system for a large petrochemical plant [21]: 20 for the expert system and 20 for the human operators.

- The above analysis comments on the number of tests. That analysis is silent on how big each single test item should be. This issue will be addressed below.

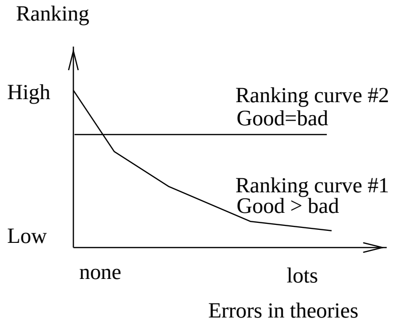

Figure 1: Two ranking curves.

3 Lower Bounds on the Size of a Single Test Item

The last section discussed how many tests to run. It was silent on the size of a single test item. Intuitively, larger theories require larger tests than smaller theories. This section explores and confirms that intuition. In the best case, an individual test case can ignore up to 90 percent of a theory.

At some critical value of U (the *U-limit*), there is not enough information in the test to check the theory. Elsewhere, we have defined a general experimental framework for finding the U-limit [20, 25, 39]. In that framework, a validation engine is executed over millions of variants to some representative theory and different values for U. The U-limit is the U value at which that validation engine cannot distinguish between good and bad theories. The U-limit can be visualised using a 2-D plot showing *theory errors* on the x-axis and *theory score* on the y-axis as in Figure 1. This visualisation is explained as follows. Suppose we can rank a theory. Intuitively, a good theory should be ranked higher than a bad theory. Ideally, if we decay from good theories to worse and worse theories, the ranking should also decay. For many applications (e.g. learning) it is also useful if:

- The ranking curve has no discontinuities.
- The ranking curve gives strong feedback if we are getting close to some ideal theory; e.g. the curve is much steeper around the good theories than bad theories.

Ranking curve 1 has the above properties. *Ranking curve 2* has none of the above properties: it is perfectly flat. *Ranking curve 2* cannot discriminate between good and bad theories. *Ranking curve 1* has been observed when testing fully measured theories (U=0), see below. However, as the percentage of unmeasured variables (U) increases, the ranking curve moves towards *ranking curve 2*. The U-limit can hence be determined by noting at what U value does the ranking curve become unacceptably flat.

Since the U-limit will be defined experimentally, the reported limit will be a domain-specific conclusion. That is, the U-limit suffers from a lack of theoretical generality. To compensate for this, a general experimental rig is described below for finding the U-limit.

```
rule1 if happy
      then motivated.
rule2 if happy and well-feed
      then lazy.
rule3 if motivated
      then not lazy.
rule4 if not lazy
      then vigilant.
```

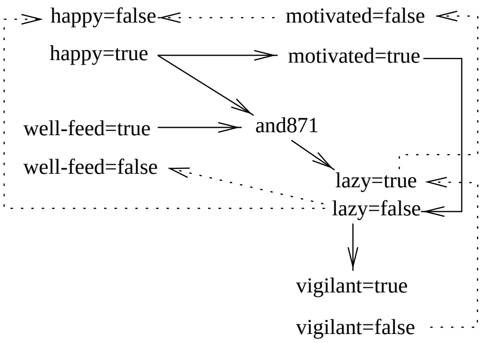

Figure 2: An example of generating `T` (right) from some rules (left). Modus tollens links shown as dashed lines.

3.1 Ranking A Theory

To draw the ranking curves, we need a validation device. This section describes HT4, the graph-based abductive validation device used in our experiments. HT4 is a generalisation and optimisation of the hypothesis testing environments of Feldman and Compton [10].

Abduction is a general framework for testing a theory [18–20,23–25,39]. Abduction is a demonstration that a theory, plus some assumptions, can reach some goal without causing contradictions [15]. If contradictions can occur, abduction must create multiple explanations. Each explanation is a maximal consistent set of beliefs which contradicts some other set of explanations. If multiple such explanations can be generated, then a BEST assessment operator selects the preferred explanation(s). HT4 uses a BEST operator that favours explanations that contain the greatest number of desired outputs. In essence, abductive validation performs the following procedure: make whatever assumptions you can to explain the greatest number of outputs.

HT4 assumes that some other program has taken some domain-specific representation and converted that into a directed and-or graph of the form `T=(V,Ed,I)`:

- Each vertex `V.i` represents the assignment of a value to a variable. `V.i` can be an AND-node or an OR-node. If an AND-node appears in a proof, then all its parents must also appear in that proof. If an OR-node appears in a proof, then one of its parents must also appear in that proof.
- Each directed edge `Ed.i` represents a statement of the possibility that the variable-value assignment in the originating node may lead to the variable-value assignment in the terminating node.
- An integrity constraint `I` complains if an illegal combination of value assignments are being made to variables. For example, `I` could complain if we tried to assign two mutually incompatible values to a variable.
- `T` is some theory generated from the user's assertion about their domain.

Examples of this `T` generation process are shown in Figure 2 and in Figure 3.

For an example of abductive validation, consider an economics theory written in our QCM language [24,38]. The model is shown in Figure 4. In QCM,

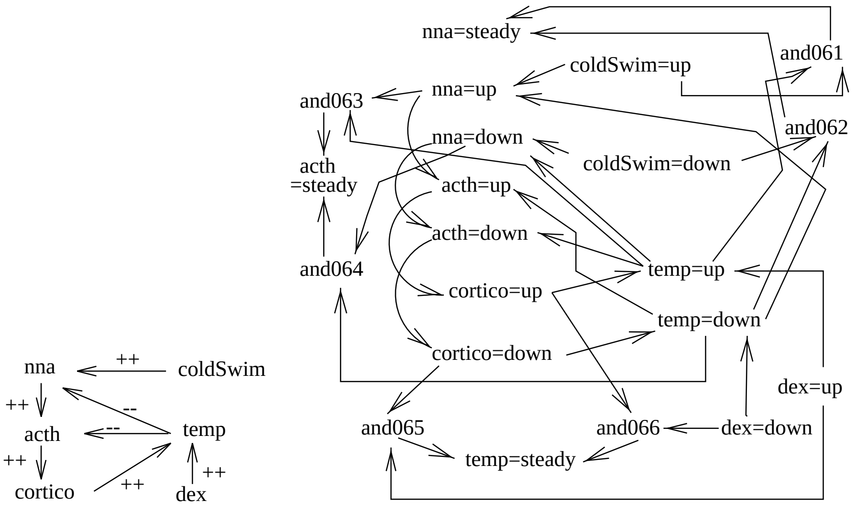

Figure 3: The Smythe'87 [36] model of stress regulation (left) and its associated theory `T` (right). In the language of the Smythe'87 theory, variables have three states: `up, down` or `steady`. Competing influences can cancel out to explain a steady. For example, note that there are two upstream influences to *nna*. Therefore, *nna* being steady can be explained by a conjunction of (e.g.) *coldSwim=up* and *temp=down*.

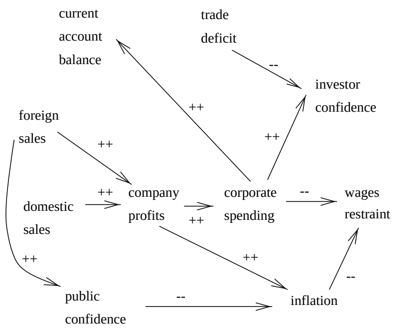

Figure 4: An economics model.

```
P.1   foreignSales=up, companyProfits=up, corporateSpending=up,
      investorConfidence=up.
P.2   domesticSales=down, companyProfits=down, corporateSpending=down
      wageRestraint=up.
P.3   domesticSales=down, companyProfits=down, inflation=down.
P.4   domesticSales=down, companyProfits=down, inflation=down,
      wagesRestraint=up.
P.5   foreignSales=up, publicConfidence=up, inflation=down.
P.6   foreignSales=up, publicConfidence=up, inflation=down,
      wageRestraint=up.
```

Table 1: Pathways that explain outputs in the economics model.

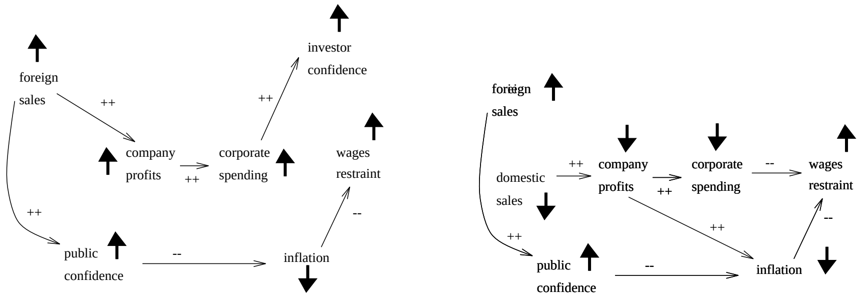

Figure 5: Explanation `E.1`, left; Explanation `E.2`, right.

theory variables have three values: *up*, *down* or *steady*. The direct connection between *foreignSales* and *companyProfits* (denoted with plus signs) means that *companyProfits* being *up* or *down* should be connected back to *foreignSales* being *up* or *down* respectively. The inverse connection between *publicConfidence* and *inflation* (denoted with minus signs) means that *inflation* being *up* or *down* should be connected back to *publicConfidence* being *down* or *up* respectively. In the case where the inputs are *(foreignSales=up, domesticSales=down)* and the output goals are *(investorConfidence=up, inflation=down, wageRestraint=up)*, then pathways can be generated to explain the outputs. as shown in Table 1. Note that some of these pathways make contradictory assumptions; e.g. *corporateSpending=up* in `P.1` and *corporateSpending=down* in `P.2`. That is, we cannot believe in `P.1` and `P.2` at the same time. If we sort these pathways into the biggest possible sets that can be believed at the same time, we arrive at the two consistent sets of explanations shown in Figure 5. Explanation `E.1` contains 3 pathways that can be believed at the same time; i.e. (`P.1, P.5, P.6`) while explanation `E.2` contains 4 pathways that can be believed at the same time; i.e. (`P.2, P.3, P.4, P.6`).

We can rank theories according to the explanations they generate. A good theory generates explanations that cover all known or desired behaviour. A bad theory cannot explain known or desired behaviour. More generally, the rank of a theory is taken from the maximum size of the intersection between its explanations and the output goals. `E.1` contains all the output goals (percent explicable=100 percent) while `E.2` contains only two of the three output goals (percent explicable=67 percent). The maximum percent explicable (a.k.a. rank) of our economics theory is the max of 100 and 67; i.e. 100 percent. It has

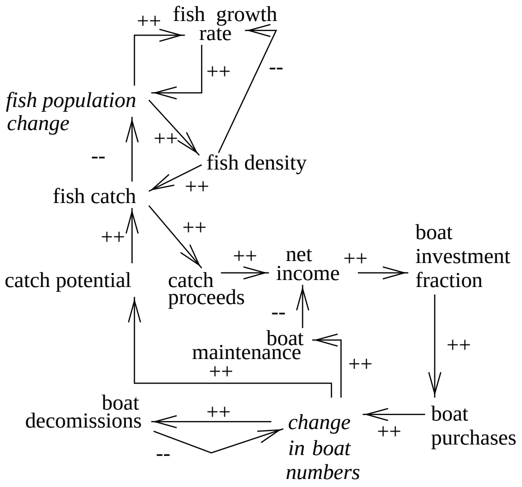

Figure 6: The fisheries model with 17 edges.

been shown that abductive validation can find bad theories (i.e. rank under 100) for many real world examples [22]. Feldman and Compton [9], followed by Menzies [24], have used this process to detect previously unseen errors in theories in neuroendocrinology published in international refereed journals. Surprisingly, these faults were found using the data published to support those theories

3.2 Introducing Errors into a Theory

The previous section lets us generate the y co-ordinates of our *U*-limit visualisation. This section lets us generate the x co-ordinates. Consider the QCM theory of fish growing in a fishery shown in Figure 6. Quantitative equations are available for this theory (see [2], pp135-141). We can run those equations to generate the correct data for the fisheries system. Once this data is available, we can generate a test with increasing *U* values by throwing away some percentage of the data, chosen at random.

Note that the fisheries theory has 17 edges and each edge has two possible annotations. We can generate theories with errors by choosing *X* percent of those edges and *corrupting* them; i.e. flipping those annotation. As *X* is increased from 0 to 100 percent, theories can be generated along the x axis of the *U*-limit visualisation.

To apply this procedure to fisheries, three of the variables were selected as inputs and *100-U* percent of the remaining variables were used as outputs. 105 sets of correct data were taken from the fisheries equations. To ensure statistical validity, between 0 and 17 randomly selected edges were corrupted 20 times (recall the above discussion on the central limit theorem). This whole process was repeated for *U* set to 0, 10, 20, ..., 90. In all, HT4 was called 105*18*20*10=378,000 times. The results are shown in Figure 7. Note that the ranking curves flatten out as we test with less and less data (e.g. the *U=90* curve is flatter than the *U=10* curve). However, even at *U=90*, the curve is far from flat. The difference between the rank given the best and worst theories

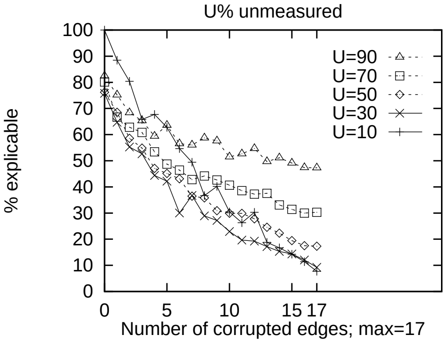

Figure 7: Looking for the "U"-limit in fisheries.

was 35 percent; i.e. it was possible to distinguish good from bad theories with up to 90 percent of the variables in fisheries ignored. Hence, no *U*-limit was found in the above experiment and theories like fisheries can be validated in the range *U=0..90.* The *U*-limit has been see in *dynamic* theories (described below). Experimentally, we know that validation for dynamic theories is defined for (best case) *U=0..70.*

3.3 Dynamic Theories

The above use of fisheries assumed a *static* interpretation of fisheries; i.e. we only generate value assignments at a single time step. A *dynamic* interpretation of fisheries allows us to generate value assignments at different time steps. To implement the dynamic interpretation, new literals are created for theory variables at time step 1, time step 2, etc. For example, *population* could spawn *population@1, population@2 ... population@T* where *T* is some time step. The invariant predicate I is extended to say that variables can have two values, but only if they assigned at different time steps.

How are we to connect literals at time *I* to literals at time *J*? Depending on how we answer this question, we can define variants on a qualitative simulation language. Eight such variants are discussed in [39], two of which are relevant to our discussion here. Consider the theory containing two edges: *direct(A,B)* and *inverse(B,A).* If we execute this theory over three time steps, we could search one of the spaces illustrated in Figure 8. In the *explicit node linking* language (or XNODE), we only cross time on the nodes explicitly denoted as time nodes by the user (in this example, *A*). In the *implicit edge linking* language (or IEDGE), we cross time on every edge. For the fisheries theory, we use the first derivative variables of *fish population change* and *change in boat numbers.* Once the search space has been defined, it can be compiled into the dependency graphs and tested using graph-based abductive validation as above.

To find the *U*-limit for IEDGE and XNODE, the ranking curves were generated as above; i.e. increasing values of *U* while corrupting 0 to 17 edges. This resulted in 105*18*20*10 calls to HT4 for XNODE, then for IEDGE (756,000 calls all together). The results are shown in Figure 9. Subjectively, we declare that:

- IEDGE becomes unacceptably flat above *U=40*; i.e. in the range *U=50..90*:

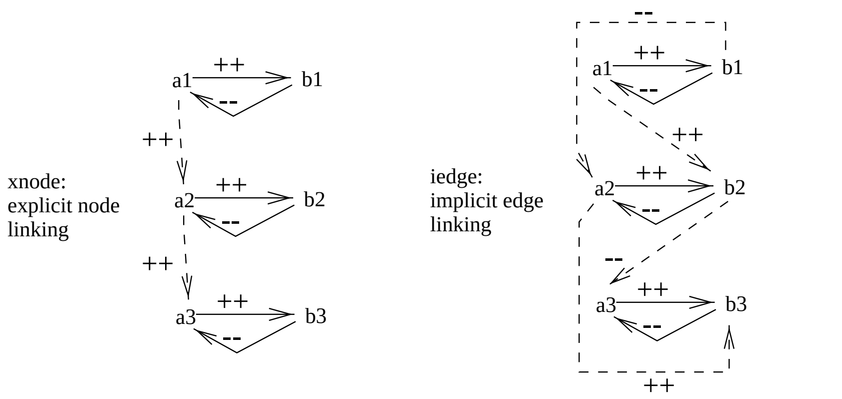

Figure 8: *Direct(A,B)* and *inverse(B,A)* renamed over 3 time intervals using the XNODE and IEDGE time linking policies. Dashed lines indicate time traversal edges.

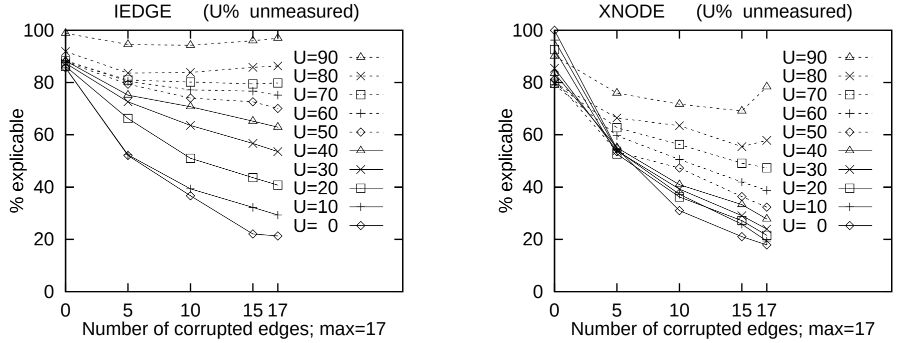

Figure 9: Looking for the *U*-limit with dynamic theories.

- It becomes very hard to distinguish good theories from bad theories.
- Testing is not practical.

- XNODE is better at large *U*. The XNODE ranking curve does not flatten out till above *U=70.* Note that this is still 20 percent worse than the *U=90* limit seen in the static fisheries experiments described above.

4 Conclusion

Testing is an important part of systems development. Humans are not perfect and neither are their artifacts. One style of testing checks if some theory can reproduce known or desired behaviour of the thing being modeled by that theory. Limits to testing can be found by considering at the limits to that style of checking. Based on that analysis, two conclusions are offered here:

- The lower practical bound on the test set size is theory independent. For each conclusion required from the system, 20 to 30 tests will suffice. Further, the work of Caraca-Valente *et.al.* suggests that for many systems, 13 tests are enough.
- The lower practical bound on the size of each test depends on the domain. The best case seen to date has been:
  - Static theories: validation is practical for *U=0..90.*
  - Dynamic theories: *U=0..70* and this is dependent on how time is interpreted within the system.

References

[1] N.M. Agnew, K.M. Ford, and P.J. Hayes. Expertise in Context: Personally Constructed, Socially elected, and Reality-Relevant? *International Journal of Expert Systems*, 7, 1 1993.

[2] H. Bossel. *Modeling and Simulations.* A.K. Peters Ltd, 1994. ISBN 1-56881-033-4.

[3] J.P. Caraca-Valente, L. Gonzalez, J.L. Morant, and J. Pozas. Knowledge-based Systems Validation: When to Stop Running Test Cases. *International Journal of Human-Computer Studies*, 1999. To appear.

[4] P.R. Cohen. *Empirical Methods for Artificial Intelligence.* MIT Press, 1995.

[5] H. S. Coles. *Thinking About the Future: A Critique of the Limits to Growth.* Sussex University Press, 1974.

[6] P. Compton, K. Horn, J.R. Quinlan, and L. Lazarus. Maintaining an Expert System. In J.R. Quinlan, editor, *Applications of Expert Systems*, pages 366–385. Addison Wesley, 1989.

[7] R.E. Courtney and D.A. Gustafson. Shotgun Correlations in Software Measures. *Software Engineering Journal*, pages 5–11, January 1983.

[8] R. Davis, H. Shrobe, and P. Szolovits. What is a Knowledge Representation? *AI Magazine*, pages 17–33, Spring 1993.

[9] B. Feldman, P. Compton, and G. Smythe. Hypothesis Testing: an Appropriate Task for Knowledge-Based Systems. In *4th AAAI-Sponsored Knowledge Acquisition for Knowledge-based Systems Workshop Banff, Canada*, 1989.

[10] B. Feldman, P. Compton, and G. Smythe. Towards Hypothesis Testing: JUSTIN, Prototype System Using Justification in Context. In *Proceedings of the Joint Australian Conference on Artificial Intelligence, AI '89*, pages 319–331, 1989.

[11] A. Ginsberg. A new Approach to Checking Knowledge Bases for Inconsistency and Redundancy. In *Proc. 3rd Annual Expert Systems in Government Conference*, pages 102–111, 1987.

[12] T.R. Gruber. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[13] W. Hamscher, L. Console, and J. DeKleer. *Readings in Model-Based Diagnosis.* Morgan Kaufmann, 1992.

[14] I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard. *Object-Oriented Software Engineering: A Use Case Driven Approach.* Addison-Wesley, 1992.

[15] A.C. Kakas, R.A. Kowalski, and F. Toni. The Role of Abduction in Logic Programming. In C.J. Hogger D.M. Gabbay and J.A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming 5*, pages 235–324. Oxford University Press, 1998.

[16] D.T. Krieger. The Hypothalmus and Neuroendocrinology. In D.T. Krieger and J.C. Hughes, editors, *Neuroendocrinology*, pages 3–122. Sinauer Associates, Inc., 1980.

[17] D.H. Meadows, D.L. Meadows, J. Randers, and W.W. Behrens. *The Limits to Growth.* Potomac Associates, 1972.

[18] T.J. Menzies. *Principles for Generalised Testing of Knowledge Bases.* PhD thesis, University of New South Wales. Avaliable from `http://www.cse.unsw.edu.au/~timm/pub/docs/95thesis.ps.gz`, 1995.

[19] T.J. Menzies. On the Practicality of Abductive Validation. In *ECAI '96*, 1996. Available from `http://www.cse.unsw.edu.au/~timm/pub/docs/96abvalid.ps.gz`.

[20] T.J. Menzies. Evaluation Issues for Problem Solving Methods, 1998. Banff KA workshop, 1998. Available from `http://www.cse.unsw.edu.au/~timm/pub/docs/97eval`.

[21] T.J. Menzies. Evaluation Issues with Critical Success Metrics. In *Banff KA '98 workshop.*, 1998. Available from `http://www.cse.unsw.EDU.AU/~timm/pub/docs/97evalcsm`.

[22] T.J. Menzies. Applications of Abduction: Knowledge Level Modeling. *International Journal of Human Computer Studies*, 45:305–355, September, 1996. Available from `http://www.cse.unsw.edu.au/~timm/pub/docs/96abkl1.ps.gz`.

[23] T.J. Menzies and R.E. Cohen. A Graph-Theoretic Optimisation of Temporal Abductive Validation. In *European Symposium on the Validation and Verification of Knowledge Based Systems, Leuven, Belgium*, 1997. Available from `http://www.cse.unsw.edu.au/~timm/pub/docs/97eurovav.ps.gz`.

[24] T.J. Menzies and P. Compton. Applications of Abduction: Hypothesis Testing of Neuroendocrinological Qualitative Compartmental Models. *Artificial Intelligence in Medicine*, 10:145–175, 1997. Available from `http://www.cse.unsw.edu.au/~timm/pub/docs/96aim.ps.gz`.

[25] T.J. Menzies and S. Waugh. Lower Limits on the Size of Test Data Sets. In *Proceedings of the Australian AI '98 conference.* World-Scientific, 1998.

[26] S. Muggleton. Inductive Logic Programming. *New Generation Computing*, 8:295–318, 1991.

[27] G.J. Myers. A Controlled Experiment in Program Testing and Code Walkthroughs/Inspections. *Communications of the ACM*, 21:760–768, 9, September 1977.

[28] A. Newell. You can't play 20 Questions with Nature, and Win. In W.G. Chase, editor, *Visual Information Processing*, pages 283–308. New York: Academic Press, 1972.

[29] J. Nielson. *Usability Engineering.* Academic Press, 1993.

[30] A.D. Preece. Principles and Practice in Verifying Rule-based Systems. *The Knowledge Engineering Review*, 7:115–141, 2 1992.

[31] A.D. Preece and R. Shinghal. Verifying Knowledge Bases by Anomaly Detection: An Experience Report. In *ECAI '92*, 1992.

[32] J.R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1:81–106, 1986.

[33] D. Sahlin. *An Automatic Partial Evaluator for Full Prolog.* PhD thesis, The Royal Institute of Technology (KTH), Stockholm, Sweden, May 1991. Available from `file://sics.se/pub/isl/papers/dan-sahlin-thesis.ps.gz`.

[34] A. TH. Schreiber, B. Wielinga, J. M. Akkermans, W. Van De Velde, and R. de Hoog. CommonKADS. A Comprehensive Methodology for KBS Development. *IEEE Expert*, 9(6):28–37, 1994.

[35] B.G. Silverman. Survey of Expert Critiquing Systems: Practical and Theoretical Frontiers. *Communications of the ACM*, 35:106–127, 4 1992.

[36] G.A. Smythe. Hypothalamic noradrenergic activation of stress-induced adrenocorticotropin (ACTH) release: Effects of acute and chronic dexamethasone pre-treatment in the rat. *Exp. Clin. Endocrinol. (Life Sci. Adv.)*, pages 141–144, 6 1987.

[37] R.E. Walpole and R.H. Myers. *Probability and Statistics for Engineers ad Scientists.* Collier Macmillion, 2 edition, 1972.

[38] S. Waugh, J. Blogs, and T. Menzies. The Temporal Qualitative Compartmental Modeling Language. In *Proceedings of the Australain AI '98 conference*, 1998.

[39] S. Waugh, T.J. Menzies, and S. Goss. Evaluating a Qualitative Reasoner. In Abdul Sattar, editor, *Advanced Topics in Artificial Intelligence: 10th Australian Joint Conference on AI.* Springer-Verlag, 1997.

[40] B.J. Wielinga, A.T. Schreiber, and J.A. Breuker. KADS: a Modeling Approach to Knowledge Engineering. *Knowledge Acquisition*, 4:1–162, 1 1992.

[41] N. Zlatereva. Truth Mainteance Systems and Their Application for Verifying Expert System Knowledge Bases. *Artificial Intelligence Review*, 6, 1992.