Average Case Coverage for Validation of AI Systems

Tim Menzies
Dept. Electrical & Computer Engineering
University of British Columbia,
Vancouver, B.C, Canada
tim@menzies.com

Bojan Cukic
Dept. Computer Science and Electrical Engineering
West Virginia University,
Morgantown, WV USA
cukic@csee.wvu.edu

Introduction

Test engineers are reluctant to certify AI systems that use nondeterministic search. The standard view in the testing field is that nondeterminism should be avoided at all costs. For example, the SE safety guru Nancy Leveson clearly states "Nondeterminism is the enemy of reliability" (Leveson 1995).

This article rejects the pessimism of the test engineers. Contrary to the conventional view, it will be argued that nondeterministic search is a satisfactory method of proving properties. Specifically, the nondeterministic construction of proof trees exhibits certain emergent stable properties. These emergent properties will allow us to rely that, on average, nondeterministic search will adequately probe the reachable parts of a search space.

Our argument assumes that properties are proved using randomized search from randomly selected inputs seeking a randomly selected goal (Menzies & Cukic 1999; 2000; Menzies *et al.* 2000; Menzies, Cukic, & Singh 2000; Menzies & Singh 2001; Menzies & Cukic 2001). Our analysis applies to any theory that can be reduced to the negation-free horn clauses of Figure 1, plus some `nogood` predicates that model incompatibilities.

Before beginning, we pause for an important caveat. This paper presents an average-case analysis of the recommended effort associated with testing. By definition, such an average case analysis says little about extreme cases of high critically. Hence, our analysis must be used with care if applied to safety-critical software.

Problems with Nondeterministic Search

When searching a nondeterministic space, an AI search engine has to make choices between incompatible options. There are two broad classes of strategies for making these choices:

1. Method one is to fork the reasoning for every possible resolution. We will call method one a *full worlds search*.
2. Method two is to pick on resolution at random, then continue on. Method two is often combined with a "rest-retry" mechanism. That is, method two is applied X times, with the system reset between each application. In the sequel, we will often refer to method two as *random worlds search*.

```
% rules
happy    if tranquillity(hi)
            or rich and healthy.
healthy  if diet(light).
satiated if diet(fatty).
tranquillity(hi) if satiated
                      or conscience(clear)

% facts
diet(fatty).
diet(light).

% contradiction knowledge
% e.g. diet(fatty) and diet(light)
% are nogood.
nogood(X,Y) :-
   X =.. [F|A1], Y =.. [F|A2], A1 \= A2.
```

Figure 1: A theory.

Both methods are problematic. Full worlds search can produce an intractably large number of possible worlds. Random worlds search can miss important conclusions. Consider the three program pathways to `happy` in Figure 2 (Figure 2 is generated from the horn clauses shown in Figure 1):

$$Path_1\ \texttt{happy} \leftarrow \texttt{tranquility}(\texttt{hi}) \leftarrow \texttt{conscience}(\texttt{clear})$$
$$Path_2 : \texttt{happy} \leftarrow \texttt{tranquility}(\texttt{hi}) \leftarrow \texttt{satiated} \leftarrow \texttt{diet}(\texttt{fatty})$$
$$Path_3 : \texttt{happy} \leftarrow \texttt{and1} \begin{cases} \leftarrow \texttt{rich} \\ \leftarrow \texttt{healthy} \leftarrow \texttt{diet}(\texttt{light}) \end{cases}$$

Some of these nodes used in these pathways are not categorical conclusions. For example, our belief in `healthy` is contingent on accepting $Path_3$ and not $Path_2$ ($Path_3$ is incompatible with $Path_2$ since these two paths require different diets). A partial random search will find only some subset of the possible paths, particularly if it is run for a heuristically selected time interval. That is, random worlds searching may not reliably infer that (e.g.) `healthy` is

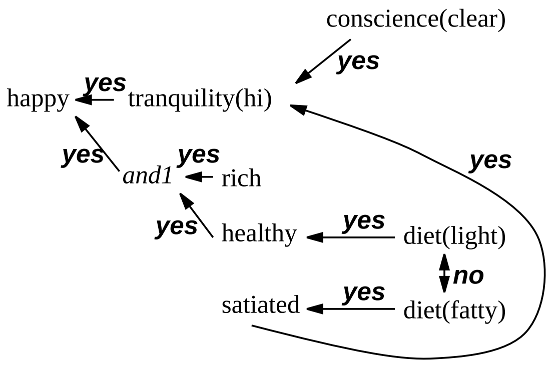

Figure 2: The rules of Figure 1 converted to an and-or graph. All nodes here are or-nodes except *and1*. All parents of an and-node must be believed if we are to believe and and-node. In this graph *no-edges* represent illegal pairs of inferences; i.e. things we can't believe at the same time such as `diet(light)` and `diet(fatty)`. All other edges are *yes-edges* which represent legal inferences.

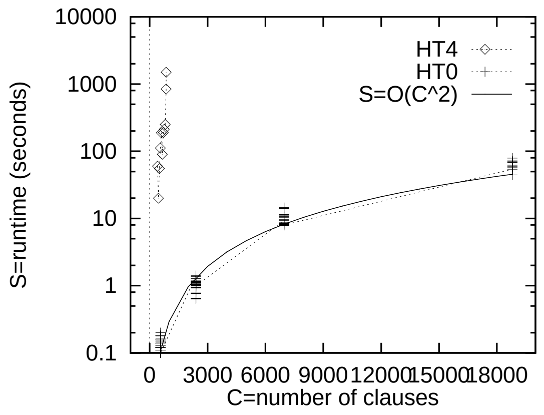

Figure 3: HT4: full worlds search search- fork one world for each consistent possibility reached from inputs. HT0: random worlds search- when contradictions are found, pick one resolution at random, then continue. In the zone where both algorithms terminated, HT0's random world search found 98% of the goals found by HT4's full worlds search.

an uncertain conclusion. Depending on how the conflict between `diet(light)` and `diet(happy)` is resolved at runtime, this system will sometimes nondeterministically conclude `healthy` and sometimes it won't.

This toy example is a simple demonstrator for a problem that can get much more complicated. A linear-time pre-processor could detect the presence of the incompatible facts `diet(light)` and `diet(happy)`. However, in larger systems, the sources of contradictions may not detectable without actually executing the system. Sadly, such executions may not reveal all the features of the theory. Gabow et.al. (Gabow, Maheshwari, & Osterweil 1976) showed that building pathways across graphs with impossible pairs (e.g.the *no-edges* in Figure 2) is NP-hard for all but the simplest graphs (a graph is very simple if it is very small, or it is a simple tree, or it has a dependency networks with out-degree ≤ 1). Hence searching graphs like Figure 2 can take exponential time and any practical proof procedure for properties across Figure 2 must use an incomplete search.

Empirical Studies of Nondeterminism

One measure of the effectiveness of a search procedure is its *cover*: the percentage of the provable goals found by a search engine. Suppose that randomized world search *covers* as many goals as a full worlds search. Such a result would be consistent with the claim that nondeterministic search is an adequate method for proving properties.

There is some empirical evidence that randomized search does indeed cover as well as full search.

- For CNF representations, it is well established that random search with retries can demonstrate satisfiability in theories too large for full search (Kautz & Selman 1996).
- Williams and Nayak found that a random worlds search algorithm performed as well as the best available ATMS system, a well-known full worlds search algorithm (Williams & Nayak 1996).
- Menzies, Easterbrook et.al. report experiments comparing random world search with full world search for requirements engineering. After millions of runs, they concluded that randomized world search found almost as many goals in less time as full worlds search (Menzies *et al.* 1999).
- In other work, Menzies and Michael compared a full worlds search search with a random worlds search. As expected, the full worlds search ran slow ($O(2^N)$) while the random worlds search ran much faster ($O(N^2)$); see Figure 3. What is more interesting is that, for problems where both search methods terminated, the random worlds search found 98% of the goals found by the full worlds search (Menzies & Michael 1999).

Theoretical Studies of Nondeterminism

The above case studies encourage a belief that nondeterminism can adequately sample a space. However, the external validity of isolated case studies must be questioned. Before we can trust nondeterministic search, we need to generalize the above case studies to show that, on average, the cover is not greatly effected by nondeterminancy.

One method of generalizing the above case studies is to consider the assumptions within the *worlds of belief* generated by search theories such as Figure 1.

- Given a model such as Figure 2 and a goal such as `happy`, heuristic search builds pathways to those goals; e.g. $Path_1 \ldots Path_3$.
- Anything that has not been asserted as a fact is an *assumption*. No path can contain mutually exclusive assumptions or contradict the goal; i.e. assuming $\neg$`happy` is illegal.

- The generated pathways should be grouped together into maximal consistent subsets called *worlds*. Our example generates two worlds: $World_1 = \{Path_1, Path_3\}$ and $World_2 = \{Path_1, Path_2\}$.
- A world contains what we can conclude from and-or graphs. A goal is proved if it can be found in a world.
- Assuming randomized search, only some of the possible paths will be found at runtime. At different runtimes, different paths and hence different worlds will be generated.

The nondeterministic testing problem can formalized as follows. The results of testing a nondeterministic system are untrustworthy when either:

1. *Not enough worlds are generated to cover the range of possible conclusions.* This first case could arise from heuristics that prune possible inferences at runtime. Such heuristics are often used when generating pathways from a space containing conflicts. Recall that search spaces like Figure 2 contain pairs of conflicts: one such pair exists for each no-edge. Recalling the above mentioned result of (Gabow, Maheshwari, & Osterweil 1976) (i.e. building pathways across programs with impossible pairs of nodes is NP-complete for all but the simplest programs), we can formally declare that any practical system must conduct an incomplete partial search of search spaces like Figure 2.
2. *Too many worlds are generated and we are swamped with possibilities.* This second case has often observed in research into qualitative reasoning. When all possible consequences of inputs are generated from a nondeterministic space, a vast suite of possibilities can be generated. Taming this intractable branching of behaviors is a major focus of research into qualitative reasoning (Menzies *et al.* 2001).

Both problems are removed if the total number of possible worlds is small. If so, then:

- All the possible conclusions can be reached by sampling just a few worlds; i.e. problem #1 goes away.
- A large number of worlds will not be possible; i.e. problem #2 goes away.

To prove that, on average, the total number of possible worlds is small, we need to categorize assumptions into three important groups. Only one of these assumption groupings will determines how many worlds are generated.

Some assumptions are *dependent* on other assumptions. For example, in $Path_3$, the `healthy` assumptions depends fully on `diet(light)`. In terms of exploring all the effects of different assumptions, we can ignore the dependent assumptions.

Another important category of assumptions are the assumptions that contradict no other assumptions. These *non-controversial* assumptions are never at odds with other assumptions and so do not effect the number of worlds generated. In our example, the non-controversial assumptions are everything except `diet(light)` and `diet(healthy)`. Hence, like the dependent assumptions, we will ignore these non-controversial assumptions.

The remaining assumptions are the *controversial, non-dependent* assumptions or *funnel* assumptions. These funnel assumptions control how all the other assumptions are grouped into worlds of belief. DeKleer's key insight in the ATMS research was that a multi-world reasoning device need only focus on the funnel (DeKleer 1986)[1]. When switching between worlds, all we need to resolve is which funnel assumptions we endorse. Continuing our example, if we endorse `diet(light)` then all the conclusions in $World_2$ follow and if we endorse `diet(healthy)` then all the conclusions in $World_1$ follow.

Paths meet and clash in the funnel. If the size of the funnel is very small, then the number of possible clashes is very small and the number of possible resolutions to those clashes is also very small. When the number of possible resolutions is very small, the number of possible worlds is very small and random search can quickly probe the different worlds of beliefs (since there are so few of them). Hence, if we can show that the average size of the funnel is small, then we can quickly poll the range of possible conclusions from our and-or graphs.

Average Funnel Size

Suppose some goal can be reached by a narrow funnel M or a wide funnel N as follows:

$$\left.\begin{array}{r} \xrightarrow{a_1} M_1 \\ \xrightarrow{a_2} M_2 \\ \ldots \\ \xrightarrow{a_m} M_m \end{array}\right\} \xrightarrow{c} goal_i \xleftarrow{d} \left\{\begin{array}{l} N_1 \xleftarrow{b_1} \\ N_2 \xleftarrow{b_2} \\ N_3 \xleftarrow{b_2} \\ N_4 \xleftarrow{b_2} \\ \quad\quad \ldots \\ N_n \xleftarrow{b_n} \end{array}\right.$$

We say that the probability of reaching the goal is the value $reached$.

Under what circumstances will the narrow funnel be favored over the wide funnel? More precisely, when are the odds of reaching $goal_i$ via the narrow funnel much greater that the odds of reaching $goal_i$ via the wide funnel? The following analysis answers those questions using the framework of (Menzies & Singh 2001) (with less needless mathematical complexity and with a wider range of distributions for the simulation studies).

To find the average funnel size, we begin with the following definitions. Let the M funnel use m variables and the N funnel use n variables. For comparison purposes, we express the size of the wider funnel as a ratio α of the narrower funnel; i.e.

$$n = \alpha m \qquad (1)$$

Each member of M is reached via a path with probability a_i while each member of N is reached via a path with probability b_i. Two paths exist from the funnels to this goal: one

[1]DeKleer called the funnel assumptions the *minimal environments*. We do not adopt that terminology here since DeKleer used consistency-based abduction while we are exploring set-covering abduction here. For an excellent discussion that defines and distinguishes set-covering from consistency-based methods, see (Console & Torasso 1991).

from the narrow neck with probability c and one from the wide neck with probability d. The probability of reaching the goal via the narrow pathway is

$$narrow = c \prod_{i=1}^{m} a_i \tag{2}$$

while the probability of reaching the goal via the wide pathway is

$$wide = d \prod_{i=1}^{n} b_i \tag{3}$$

Let $P(narrow|reached)$ and $P(wide|reached)$ denote the conditional probabilities of using one of the funnels, given that the goal is reached. The ratio R of these conditional probabilities informs us when the narrow funnel is favored over the wider funnel.

$$R = \frac{P(narrow|reached)}{P(wide|reached)} = \frac{\left(\frac{narrow}{reached}\right)}{\left(\frac{wide}{reached}\right)} = \frac{narrow}{wide} \tag{4}$$

Narrow funnels are more likely than wider funnels when $R > 1$.

To compute the frequency of $R > 1$, we have to make some assumptions about the probability distributions of $narrow$ and $reached$. (Menzies & Singh 2001) showed that if a_i and b_i come from uniform probability distributions, then narrow funnels are more likely than wide funnels. In the case of such uniform distributions,

$$\sum_{i=1}^{m} a_i = 1 \therefore a_i = \frac{1}{m} \therefore narrow = c\left(\frac{1}{m}\right)^m \tag{5}$$

Similarly, under the same assumptions,

$$wide = d\left(\frac{1}{n}\right)^n \tag{6}$$

Under this assumption of uniformity, $R > 1$ when

$$\frac{narrow}{wide} = \frac{c\left(\frac{1}{m}\right)^m}{d\left(\frac{1}{n}\right)^n}$$

Recalling that $n = \alpha m$, this expression becomes

$$(\alpha m)^{\alpha m} m^{-m} > \frac{d}{c} \tag{7}$$

Consider the case of two funnels, one twice as big as the other; i.e. $\alpha = 2$. This expression can then be rearranged to show that $\frac{narrow}{wide} > 1$ is true when

$$(4m)^m > \frac{d}{c} \tag{8}$$

At $m = 2$, Equation 8 becomes $d < 64c$. That is, to access $goal_i$ from the wider funnel, the pathway d must be 64 times more likely than the pathway c. This is not highly likely and this becomes less likely as the narrower funnel grows. By the same reasoning, at $m = 3$, to access $goal_i$ from the wider funnel, the pathway d must be 1728 times more likely than the narrower pathway c. That is, under the assumptions of this uniform case, as the wide funnel gets wider, it becomes less and less likely that it will be used.

To explore the case where $\sum_{i=1}^{m} a_i \neq 1$ and $\sum_{i=1}^{m} b_i \neq 1$ (i.e. the non-uniform probability distribution case), we created and executed a small simulator many times. In this simulator, we found the frequency at which $R > t$ where t was some threshold value.

To execute the simulator, we required some knowledge of the distributions of $narrow$ and $wide$ when they are computed by nondeterministic search.Those distributions were taken from an average case analysis of reachability across graphs such as Figure 2. This reachability analysis is discussed below.

A Reachability Model

Menzies, Cukic, Singh and Powell (Menzies *et al.* 2000) computed the odds of reaching some random part of a space of nondeterminant choices from random inputs. The analysis assumed that software had been transformed into a possibly cyclic directed graph containing and-nodes and or-nodes; e.g. Figure 1 has been converted to Figure 2. A simplified description of their analysis is presented here. For full details, including details such as using random variables and testing for loops and contradictions, see (Menzies *et al.* 2000).

Assume that "in" number of inputs have been presented to a graph containing V nodes. From these inputs, we grow a tree of pathways down to some random node within the graph. The odds of reaching a node straight away from the inputs is

$$x_0 = \frac{in}{V} \tag{9}$$

The probability of reaching an and-node with $andp$ parents is the probability of reaching all its parents; i.e.

$$x_{and} = x_i^{andp} \tag{10}$$

where x_i is the probability we computed in the prior step of the simulation (and x_0 being the base case). The probability of reaching an or-node with orp parents is the probability of not missing any of its parents; i.e.:

$$x_{or} = 1 - (1 - x_i)^{orp} \tag{11}$$

If the ratio of and-nodes in a network is $andf$, then the ratio of or-nodes in the same network is $1 - andf$. The odds of reaching some random node x_j is the weighted sum of the probabilities of reaching and-nodes or or-nodes; i.e.

$$x_j = andf * x_{and} + orf * x_{or} \tag{12}$$

We can convert x_j to the number of tests N required to be 99% sure of find a fault with probability x_j as follows. Equation 11 is really the sampling-with-replacement equation where orp is the number of trials N. We can use sampling-with-replacement to find the certainty of finding some event after N trials. If we demand a 99% certainty of reaching a node at step j (i.e. $y = 0.99$), then we can re-arrange Equation 11 to

$$N = \frac{log(1 - 0.99)}{log(1 - x_j)} \tag{13}$$

After 150,000 simulations of this model, some best and worst cases were identified. These are shown in Figure 4 labeled *pessimistic* and *optimistic* respectively. In the pessimistic case, we restricted the depth of our search to some trivial size: $j < 10$. In this pessimistic case, more than 10,000 random inputs are required to reach half the nodes in the graphs we simulated. In the optimistic case, we gave the search engine greater freedom to explore: $j < 100$. In this optimistic case, less than 100 random inputs would reach over half the nodes in the graphs we simulated.

Simulating Funnels

Having some knowledge of the distributions, we can now compute the frequency of $R > t$ for non-uniform distributions. For one run of the Equation 4 simulator, m and α were picked at random from the ranges:

$$m \in \{1, 2, \dots 10\}; \quad \alpha \in \{1, 1.25, 1.5, \dots 10\}$$

a_i, b_i, c, d were taken from one of three distributions: the pessimistic and optimistic curves shown in Figure 4, plus a log normal curve (just for comparison purposes). For the log-normal curve, mean μ and standard deviation σ^2 of the logarithm of the variables were picked at random from the following ranges:

$$\mu \in \{1, 2, \dots 10\}; \quad spread \in \{0.05, 0.1, 0.2, 0.4, 0.8\}$$

μ and $spread$ where then converted into probability as follows:

$$\sigma^2 = spread * \mu; \quad probability = 10^{-1*normDist(\mu,\sigma^2)}$$

R was then calculated and the number of times R exceeded different values for t is shown in Figure 5. As might be expected, at $t = 1, \alpha = 1$ the funnels are the same size and the odds of using one of them is 50%. As α increases, then increasingly $R > t$ is satisfied and the narrower funnel is preferred to the wider funnel. The effect is quite pronounced.

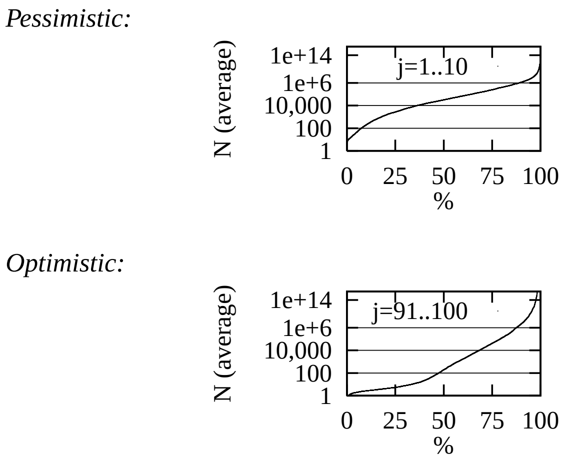

Figure 4: 150,000 runs of the simulator generated x_j figures which were converted into number of tests N required using Equation 13. X-axis shows the percentile distribution of the output of the runs.

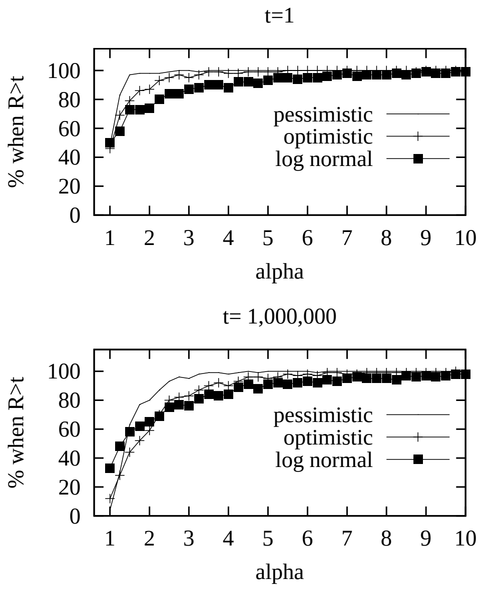

Figure 5: 10000 runs of the funnel simulator. Y-axis shows what percentage of the runs satisfies $R > t$.

For example, for all the studied distributions, if the wider funnel is 2.25 times bigger than the narrow funnel, random search will be 1,000,000 times as likely as to use the narrow funnel (see the lower graph of Figure 5). Interestingly, as reachability drops, the odds of using the narrow funnel increase (see the *pessimistic curves* in Figure 5). That is, the harder the search, the less likely the search will suffer from the problem of indeterminate search under-sampling the space.

Discussion

Why have these optimistic average case results for nondeterministic search been reported before? Several research communities might have seen the same results. However, the premises of those communities may have blocked them from finding these conclusions:

1. The formal verification community seeks near-complete inference procedures Our results assume random search and random search is not attractive to proponents of complete search.
2. CNF-based theorem provers do not have a strong concept of a proof tree. Hence, the satisfiability community would not have seen the above results since our analysis requires knowledge of the traversal path taken from inputs to goals.
3. The software engineering (SE) testing community uses procedural systems that take fixed pathways through their systems. Our results assume that a test procedure can take random detours during the test procedure. This is not an acceptable procedure for most of the SE test community. However, amongst the small group of SE test researchers that conduct randomization studies of SE sys-

tem, some recent empirical results endorse our proposal that random search is an adequate test strategy (Horgan & Mathur 1996).

Conclusion

It has been argued that for a wide range of distributions (uniform, pessimistic, optimistic, log-normal), random search engines will favor worlds with narrow funnels. As funnel size grows, the number of different worlds also grows. Hence, size the average funnel size is small, the number of different worlds will also be small. Consequently:

- On average, the impact on goal coverage due to nondeterminism is small.
- Nondeterministic search is an adequate method for proving properties in a system.

Acknowledgments

Holgar Hoss's comments on the maths of (Menzies & Singh 2001) prompted the simpler analysis shown here.

References

Console, L., and Torasso, P. 1991. A Spectrum of Definitions of Model-Based Diagnosis. *Computational Intelligence* 7:133–141.

DeKleer, J. 1986. An Assumption-Based TMS. *Artificial Intelligence* 28:163–196.

Gabow, H.; Maheshwari, S.; and Osterweil, L. 1976. On two problems in the generation of program test paths. *IEEE Trans. Software Engrg* SE-2:227–231.

Horgan, J., and Mathur, A. 1996. Software testing and reliability. In Lyu, M. R., ed., *The Handbook of Software Reliability Engineering*, 531–565.

Kautz, H., and Selman, B. 1996. Pushing the envelope: Planning, propositional logic and stochastic search. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, 1194–1201. Menlo Park: AAAI Press / MIT Press. Available from `http://www.cc.gatech.edu/~jimmyd/summaries/kautz1996.ps`.

Leveson, N. 1995. *Safeware System Safety And Computers*. Addison-Wesley.

Menzies, T., and Cukic, B. 1999. On the sufficiency of limited testing for knowledge based systems. In *The Eleventh IEEE International Conference on Tools with Artificial Intelligence. November 9-11, 1999. Chicago IL USA.*

Menzies, T., and Cukic, B. 2000. When to test less. *IEEE Software* 17(5):107–112. Available from `http://tim.menzies.com/pdf/00iesoft.pdf`.

Menzies, T., and Cukic, B. 2001. *Submitted to Handbook of Software Engineering and Knowledge Engineering, Volume II.* World-Scientific. chapter How Many Tests are Enough? Available from `http://tim.menzies.com/pdf/00ntests.pdf`.

Menzies, T., and Michael, C. 1999. Fewer slices of pie: Optimising mutation testing via abduction. In *SEKE '99, June 17-19, Kaiserslautern, Germany. Available from `http://tim.menzies.com/pdf/99seke.pdf`.*

Menzies, T., and Singh, H. 2001. Many maybes mean (mostly) the same thing. In *Submitted to the 2nd International Workshop on Soft Computing applied to Software Engineering (Netherlands), February*. Available from `http://tim.menzies.com/pdf/00maybe.pdf`.

Menzies, T.; Easterbrook, S.; Nuseibeh, B.; and Waugh, S. 1999. An empirical investigation of multiple viewpoint reasoning in requirements engineering. In *RE '99*. Available from `http://tim.menzies.com/pdf/99re.pdf`.

Menzies, T.; Cukic, B.; Singh, H.; and Powell, J. 2000. Testing nondeterminate systems. In *ISSRE 2000*. Available from `http://tim.menzies.com/pdf/00issre.pdf`.

Menzies, T.; Cohen, R.; Waugh, S.; and Goss, S. 2001. Applications of abduction: Testing very long qualitative simulations. *IEEE Transactions of Data and Knowledge Engineering (to appear)*. Available from `http://tim.menzies.com/pdf/97iedge.pdf`.

Menzies, T.; Cukic, B.; and Singh, H. 2000. Agents talking faster. NASA Goddard Workshop on Formal Aspects of Agent-Oriented Systems. Available from `http://tim.menzies.com/pdf/00godd.pdf`.

Williams, B., and Nayak, P. 1996. A model-based approach to reactive self-configuring systems. In *Proceedings, AAAI '96*, 971–978.