Model Abstraction for Testing of Physical Systems

Peter Struss
Technical University of Munich
Computer Science Dept.
Orleansstr. 34
D-81667 Munich
Germany
struss@informatik.tu-muenchen.de

Abstract

We present a formal theory of model-based testing and an algorithm for test generation based on it, and outline how testing is implemented by a diagnostic engine. The key to making the complex task of test generation feasible for systems with continuous domains is the use of model abstraction. Tests can be generated using manageable finite models and then mapped back to a detailed level. We state conditions for the correctness of this approach and discuss the preconditions and scope of applicability of the theory.

1 Introduction

Testing means shifting a system into different states by appropriate inputs in order to find observations that determine its present behavior mode. Often, the tests are designed to *confirm a particular behavior,* usually the correct or intended one, for instance in manufacturing. In diagnosis we may, in contrast, want discriminating tests which effectively and efficiently *identify the present* (faulty) *behavior.* This paper focuses on confirming tests. There exist theories and algorithms for test generation in particular domains. For digital circuits, for instance, a solution is feasible because, although the number of components can be large, the individual components exhibit a simple behavior and, more fundamentally, because of the Boolean domain of the variables ((Roth 1980), (Gupta & Welham 1989), (Camurati et al. 1990)). For variables with large domains or for physical systems with continuous behavior, these techniques are not applicable. In extending methods from model-based diagnosis, and exploiting our work on multiple modeling (Struss 1992), we propose a general theory that addresses the generation and application of tests in such domains.

We first discuss the problems addressed and outline the basic ideas of our approach by presenting a simple (continuous and dynamic) system, a thyristor. In section 3, we present the basic theory and an algorithm for test generation. Testing of constituents in the context of a whole device is shown to be a straightforward extension in section 4. Section 5 outlines briefly how testing is implemented by a standard model-based diagnosis engine. Finally, we discuss the achievements, preconditions, and restrictions of the approach.

Due to space limitations, we do not always treat the most general cases, and we omit proofs. Both can be found in the long version of this paper (Struss 1994).

2 The Intuition behind Testing

In the following, we consider a continuous dynamic system as an illustrative example (rather than a serious application). A thyristor is a semi-conductor with anode, A, cathode, C, and gate, G, that operates as a (directed) switch: it works in two states, either conducting current in a specified direction with almost zero resistance (exaggerated by the upper line of the simplified characteristic curve in Fig. 2.1a), or blocking current like a resistor with almost infinite resistance (the horizontal line). The transition from the OFF state to ON is controlled by the gate; if it receives a pulse the thyristor "fires", provided the voltage drop exceeds a threshold, V_{Th}. There is a second way to fire a thyristor (which is normally avoided, but may occur in certain circuits and situations), namely if the voltage drop exceeds the breakover voltage, V_{Bo} as is indicated by the characteristic in Fig. 2.1a. The annotation with 1 and 0 indicates the presence and absence of a gate pulse. So, for instance, for $\Delta V > V_{Bo}$ the thyristor is ON (i>0), no matter whether or not it receives a gate pulse and, hence, the annotation with 0 and 1. In contrast, the section $V_{Th} < \Delta V < V_{Bo}$, i>0 is annotated with 1, because a gate pulse is required for firing. This representation is based on the assumption that switching happens instantaneously (turn-On time and spreading time 0).

Now suppose we want to test a thyristor, i.e. to make sure that it behaves according to the described correct behavior. This creates several problems: voltage and current are considerered to have a continuous domain. We can only gather a finite set of sample observations. But if they all agree with the desired behavior, what would then make us confident that more observations could not reveal a contradiction to this behavior? It is the fact that *there is no other possible behavior* (a faulty one) *that would also be consistent with the previous observations.*

What are the possible faults of a thyristor? A thyristor may be *punctured*, i.e. acting like a wire, or *blocking* like an open switch. A third kind of fault may be due to the fact that the actual *breakover voltage is less than the*

nominal one, with the result that the thyristor fires at a voltage drop well below V_{Bo} without a gate pulse. With V'_{Bo} we denote the lowest tolerable actual breakover voltage (or the highest one which is considered to characterize a faulty behavior). Fig. 2.1 shows the (idealized) characteristics of these behaviors in comparison to the correct behavior.

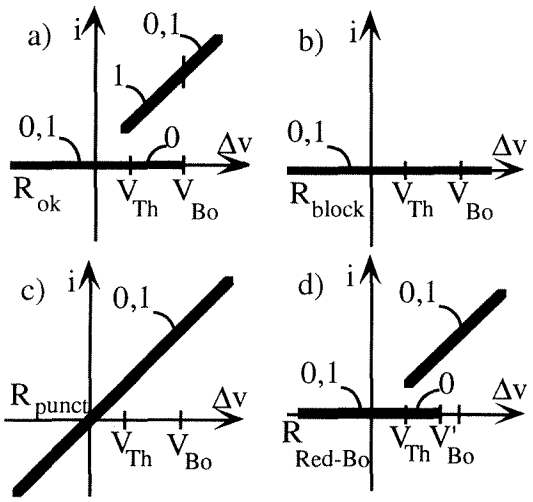

Figure 2.1 The characteristics of the behaviors of a thyristor: a) correct b) blocking c) punctured d) with a reduced breakover voltage

Considering these behaviors (and, perhaps, looking at the figures), we may get the following idea for a set of two tests: the first one with a high voltage drop (i.e. between V'_{Bo} and V_{Bo}) without a gate pulse, and a second one with a medium or high voltage drop (i.e. between V_{Th} and V_{Bo}) in conjunction with a gate pulse. If we obtain results that comply with the correct behavior in both cases (zero current for the former, positive current for the latter), then the thyristor must be correct, because these observations rule out all three types of faults: the first one contradicts the punctured behavior and a reduced breakover voltage, while the second one refutes the blocking mode. This simple example illustrates several fundamental ideas :

- A particular behavior is confirmed if all others can be refuted by some finite set of observations.
- We obtain such sets of tests by describing behaviors through relations among variables and by determining their distinctions (i.e. set differences).
- We may end up with less tests than the number of behaviors to be refuted (in the thyristor example two tests for an *infinite* number of behaviors).

Finally, the thyristor indicates a way to address the complexity problem when we have to handle large or even infinite domains:

- We may be able to perform test generation using a (qualitative) abstraction of the behavior description (e.g. with characterizations such as "high" and "medium").

In the remainder of this paper we develop these ideas into a formal theory and an algorithmic solution for test generation and testing.

3 Test Generation for Single Constituents

First, we present the basic definitions and results that allow the generation of tests, based on relational behavior models. For all definitions and theorems, we first paraphrase them in English before presenting the formal statement. Throughout this section, we consider one constituent (component, mechanism, process, subsystem that is) of a system that is assumed to be accessible. It has a (not necessarily finite) set of possible, mutually exclusive behaviors, BEHVS, associated with it. This set is assumed to be *exhaustive*, i.e. the constituent has exactly one behavior $B_i \in$ BEHVS. Later, we will discuss the case of this assumption being wrong. That the constituent has one unique behavior seems to exclude the possibility of intermittent behaviors. In (Struss 94a) we show that this is not the case.

3.1 The Foundation: Finding Observable Distinctions

As motivated by the example (and common in model-based reasoning systems which use constraints for modeling), we describe behavior modes by the set of value tuples that are possible under this behavior, i.e. by a relation R in some representation. Using the formalism of (Struss 1992) such a representation is determined by selecting a vector

$$\underline{v} = (v_1,...,v_2)$$

of local variables and their respective domains:

$$DOM(\underline{v}) = DOM(v_1) \times DOM(v_2) \times .... \times DOM(v_k).$$

For the time being, we assume one fixed representation ($\underline{v}$, DOM($\underline{v}$)), because this simplifies the notation and is not an essential restriction (the general case is treated in (Struss 1994)). The behavior models of the thyristor can be described in the representation

$$(\underline{v}_{Th}, DOM(\underline{v}_{Th})) = ((\Delta V, gate, i), \mathbb{R}\times\{0,1\}\times\mathbb{R}).$$

The relations $R \subset DOM(\underline{v}_{Th})$ from Fig. 2.1 modeling the thyristor behaviors are shown in Table 3.1. The inevitable inaccuracy in this model is reflected by the (small) numbers δ and Δ.By SIT we denote the set of situations physically possible under the present mode of a constituent. We define a behavior model M(R) as the claim that the relation R covers all value tuples $\underline{v}$ may take in a situation $s \in$ SIT:

Definition 3.1 (Behavior Model)

$$M(R) : \Leftrightarrow$$

$$\forall \underline{v}_0 \in DOM(\underline{v})\ (\exists s \in SIT\ \ \underline{v}(s)=\underline{v}_0) \Rightarrow \underline{v}_0 \in R$$ [1]

[1] $\underline{v}(s) = \underline{v}_0$ means that $\underline{v}$ has the value $\underline{v}_0$ in situation s rather than equality. Because $\underline{v}$ can take different values

	ΔV	gate	i
R_{ok}			
	$(-\infty, 0]$	{0,1}	$[-\delta, 0]$
	$(0, V_{Th}]$	{0,1}	$[0, \delta]$
	$(V_{Th}, V_{Bo}]$	{0}	$[0, \delta]$
	$(V_{Th}, V_{Bo}]$	{1}	$[(res-\Delta)*\Delta V,(res+\Delta)*\Delta V]$
	(V_{Bo}, ∞)	{0,1}	$[(res-\Delta)*\Delta V,(res+\Delta)*\Delta V]$
R_{block}			
	$(-\infty, 0]$	{0,1}	$[-\delta, 0]$
	$(0, \infty)$	{0,1}	$[0, \delta]$
R_{punct}			
	$(-\infty, \infty)$	{0,1}	$[(res-\Delta)*\Delta V,(res+\Delta)*\Delta V]$
R_{Red-Bo}			
	$(-\infty, 0]$	{0,1}	$[-\delta, 0]$
	$(0, V_{Th}]$	{0,1}	$[0, \delta]$
	$(V_{Th}, V'_{Bo}]$	{0}	$[0, \delta]$
	(V_{Th}, ∞)	{0,1}	$[(res-\Delta)*\Delta V,(res+\Delta)*\Delta V]$

Table 3.1 Relations modeling thyristor behaviors. Each is the union of the lines of the table; e.g. the first line of R_{ok} is to be read $(-\infty, 0] \times [-\delta, 0] \times \{0,1\}$.

If $M(R_i)$ is a model of the behavior $B_i \in BEHVS$, i.e.

$B_i \Rightarrow M(R_i)$,

and if an observation (obs) contradicts the behavior model, i.e. lies outside R_i, then we can safely rule out the behavior:

$obs \Rightarrow \neg M(R_i) \vdash\!\!\!- \; obs \Rightarrow \neg B_i$.

While this provides a way for *refuting* behaviors, we are interested in *confirming* a particular behavior.

As suggested by the example, tests are defined as sets of value tuples such that observing at least one tuple in each set in reality allows us to conclude the presence of a behavior mode. More formally: a set of value tuples $V = \{\underline{v}_i\}$ containing at least one tuple out of each T_i,

$\forall T_i \; \exists \underline{v}_i \in V \quad \underline{v}_i \in T_i$,

is called a *hitting set* of $\{T_i\}$. The fact that all the values in V are actually taken in some real situation is denoted by the sentence φ_V:

$\varphi_V \equiv \forall \underline{v}_i \in V \; \exists s_i \in SIT \; \underline{v}(s_i) = \underline{v}_i$.

Definition 3.2 (Test, Confirming Test Set)

A test is a non-empty relation on some representational space: $T_i \subseteq DOM(\underline{v})$.

A set $\{T_i\}$ of tests is a confirming test set for a behavior $B_0 \in BEHVS$ iff for all hitting sets V of $\{T_i\}$, observation of V entails B_0:

$\varphi_V \vdash\!\!\!- B_0$.

What assured us that the tests in section 2 actually confirm the thyristor's correct behavior? The fact that no other behavior mode would survive observations from both tests. In general, fo each behavior B_j, different from the one to be confirmed, there must exist a test T_i lying completely outside a modeling relation of B_j. In other words, the complement of T_i,

$T_i^c := DOM(\underline{v}) \backslash T_i$,

specifies a model of B_j. This is stated by Lemma 3.1.

Lemma 3.1

$\{T_i\}$ is a confirming test set for B_0 iff

$\forall B_j \in BEHVS \; B_j \neq B_0 \Rightarrow (\exists T_i \quad B_j \Rightarrow M(T_i^c))$.

A test is only useful if it is observable. So, in the following, let $OBS(\underline{v}) \subseteq VARS(\underline{v})$ be the set of observable variables in the representation $(\underline{v}, DOM(\underline{v}))$ with the respective projection (see Fig. 3.4)

$p_{obs} : DOM(\underline{v}) \rightarrow DOM(\underline{v}_{obs})$.

Definition 3.3 (Observable Test Set)

A test set $\{T_i\}$ is observable, if all T_i are observable, i.e. $T_i \subset DOM(\underline{v}_{obs})$.

Lemma 3.1 indicates the way to generate confirming (observable) test sets for some behavior $B_0 \in BEHVS$: we have to find (observable) distinctions between B_0 and each other mode B_i, and confirm these distinctions to be present. We can grasp them as the set differences

$D_i := p_{obs}(R_0) \backslash p_{obs}(R_i)$

of appropriate modeling relations of these behaviors. The number of differences D_i can be smaller than the number of behaviors to be refuted, because the modeling relations chosen may cover several behaviors (For the thyristor, for instance, R_{RED-Bo} covers an infinite set of behaviors).

We even do not have to enumerate all behaviors in BEHVS, and we do not have to be able to describe them in detail; we only have to be sure that the set of relations $\{R_i\}$ covers all possible behaviors. Table 3.2 and Fig. 3.1 show the set differences obtained for the thyristor example from the relations in Table 3.1 (in our case $OBS(\underline{v}_{Th}) = VARS(\underline{v}_{Th})$ holds, i.e. all variables are considered observable).

	ΔV	gate	i
D_{block}			
	$(V_{Th}, V_{Bo}]$	{1}	$[(res-\Delta)*\Delta V,(res+\Delta)*\Delta V]$
	(V_{Bo}, ∞)	{0,1}	$[(res-\Delta)*\Delta V,(res+\Delta)*\Delta V]$
D_{punct}			
	$(-\infty, 0]$	{0,1}	$[0, \delta]$
	$(0, V_{Th}]$	{0,1}	$[0, \delta]$
	$(V_{Th}, V_{Bo}]$	{0}	$[0, \delta]$
D_{Red-Bo}			
	$(V'_{Bo}, V_{Bo}]$	{0}	$[0, \delta]$

Table 3.2 The differences between the relation characterizing the correct behavior and the fault mode relations

(from different domains, but also in the same domain), (Struss 1992) uses a special predicate Val .

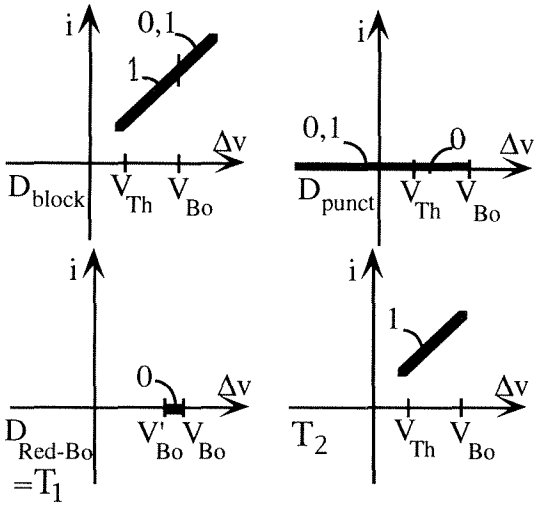

Figure 3.1 The relations of Table 3.1 and the tests of Table 3.3

Any observable test refuting $M(R_i)$ and containing only tuples consistent with $M(R_0)$ must be a subset of D_i. Although we could use $\{D_i\}$ as a test set, we may further reduce the number of tests by replacing several D_i by a common subset. We call a set of sets, $\{T_k\}$, a *hitting set of sets* of $\{D_i\}$, if it contains a non-empty subset of each D_i:

$$\forall D_i \ \exists T_k \ \emptyset \neq T_k \subseteq D_i \ .$$

The following lemma is the basis for the generation of observable confirming test sets:

Lemma 3.2

Let

$\{R_i \mid R_i \subset DOM(\underline{v})\}$

cover all behaviors (except B_0):

$\forall B_j \in BEHVS \backslash \{B_0\} \ \exists R_i \ B_j \Rightarrow M(R_i)$,

and $R_0 \subset DOM(\underline{v})$ cover B_0:

$B_0 \Rightarrow M(R_0)$.

If $\{T_k\}$ is a hitting set of sets of

$\{D_i\} := \{p_{obs}(R_0) \backslash p_{obs}(R_i)\}$,

then it is an observable confirming test set for B_0.

The thyristor test set is an illustration of Lemma 3.2. Since $D_{Red\text{-}Bo} \subset D_{punct}$, the set of relations

$\{D_{block}, D_{Red\text{-}Bo}\}$

is a hitting set of sets of

$\{D_{block}, D_{punct}, D_{Red\text{-}Bo}\}$

and forms an observable confirming test set for the correct behavior.

We also obtain a neccessary condition for the existence of a confirming test set: if B_0 is actually a restriction of some other behavior B_j, it is impossible to find a confirming test set for B_0. An example for this case is an intermittent fault which is characterized by a relation that covers the correct behavior entirely (because sometimes the constituent behaves correctly). This is intuitive, because even if we observe only value tuples consistent with the correct behavior, so far, we can never be sure that the future will not reveal contradictory observations (whenever the fault occurs). Note that even if $R_0 \backslash R_i$ is non-empty, D_i may be empty, because the distinction is not observable in the given representation.

Now we have determined test sets that confirm a particular behavior, *if* they are observed. However, we do not want to wait for them to drop from heaven, but we would like to enforce them by an appropriate *causal input* to the system.

3.2 Finding Deterministic Test Inputs

We assume that the causal variables are observable, which is reasonable, because it means we know what we are doing to the constituent. So, let

$$CAUSE(\underline{v}) \subseteq OBS(\underline{v}) \subseteq VARS(\underline{v})$$

be the set of susceptible variables and

$$p_{cause} : DOM(\underline{v}) \rightarrow DOM(\underline{v}_{cause})$$

$$p'_{cause} : DOM(\underline{v}_{obs}) \rightarrow DOM(\underline{v}_{cause})$$

the respective projections into the set of input tuples (see Fig. 3.4). What we would like to have is test inputs, i.e. subsets of $DOM(\underline{v}_{cause})$, that are guaranteed to determine whether or not a particular behavior is present. More precisely: if we input one tuple out of each set to the constituent, the resulting value tuples of $\underline{v}$ deterministically either confirm or refute the behavior:

Definition 3.4 (Test Input, Deterministic Input Set)

A test input is a non-empty relation on $DOM(\underline{v}_{cause})$:

$TI_i \subseteq DOM(\underline{v}_{cause})$.

A set of test inputs $\{TI_i\}$ is deterministic for a behavior $B_0 \in BEHVS$ iff for all sets

$V = \{\underline{v}_i\} \subseteq DOM(\underline{v})$

whose set of causes $\{p_{cause}(\underline{v}_i)\}$ forms a hitting set of $\{TI_i\}$, observation of V is inconsistent with B_0 or it entails it:

$\varphi_v \vdash \neg B_0 \quad or \quad \varphi_v \vdash B_0$

How can we generate deterministic input sets? Unfortunately, for a test set $\{T_i\}$ confirming B_0, the input set $\{p_{cause}(T_i)\}$ is not necessarily deterministic.

To illustrate this, we consider the relation R_{neg} which is a subset of $R_{ok} \backslash R_{punct}$ (for $\Delta V < 0$) and which could be used to rule out the fault "punctured" of the thyristor (Fig. 3.2). p_{cause} projects to (ΔV, gate):

$$p_{cause}(R_{neg}) = (-\infty,-\varepsilon) \times \{0\}.$$

However, if we choose a test input with (ΔV, gate) out of $(-\infty,-\varepsilon) \times \{0\}$, a value of i might be observed such that the vector lies in the intersection of R_{ok} and R_{punct} (indicated by "×" in Fig. 3.2) and, hence, is consistent with the correct behavior but also fails to refute the fault. As a cure, we have to exclude $p_{cause}(R_{ok} \cap R_{punct})$, i.e. to reduce the test input for Δv to $(-\infty, \varepsilon')$.

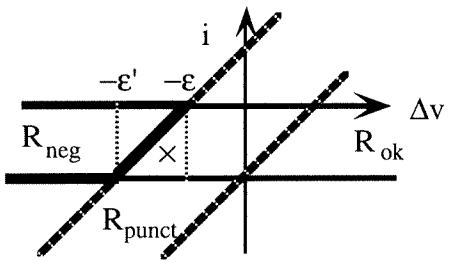

Figure 3.2 $p_{cause}(R_{ok} \setminus R_{punct})$ and $p_{cause}(R_{ok} \cap R_{punct})$ overlap

More generally, in order to construct input sets deterministic for some $B_0 \in$ BEHVS and leading to observable test sets, for each $B_i \neq B_0$ we have to determine and eliminate those inputs that possibly lead to the same observations under both B_0 and B_i. This is the set

$p'_{cause}(p_{obs}(R_0) \cap p_{obs}(R_i))$.

Hence, if we define

$DI_i := p_{cause}(R_0) \setminus p'_{cause}(p_{obs}(R_0) \cap p_{obs}(R_i))$,

then we are guaranteed that any input chosen from DI_i causes an observable value tuple that is inconsistent with $M(R_i)$ or with $M(R_0)$ (possibly with both of them). This is the idea underlying the proof of Theorem 3.3.

Theorem 3.3

Under the conditions of Lemma 3.2, each set of test inputs $\{TI_k\}$ that is a hitting set of sets of

$\{DI_i\} = p_{cause}(R_0) \setminus p'_{cause}(p_{obs}(R_0) \cap p_{obs}(R_i))$

is deterministic for B_0 and

$\{T_k\} := \{p_{obs}(R_0) \cap p'^{-1}_{cause}(TI_k)\}$

is an observable confirming test set for B_0.

In practice, one wants to avoid test inputs that are extreme and possibly cause (or make worse) damage. For instance, we do not want to test with $\Delta V > V_{B0}$, because the thyristor could be destroyed. In this case, DI_i may have to be further reduced by intersecting it with a set of admissible inputs:

$DI_{iadm} := R_{adm} \cap DI_i$.

For the thyristor, we choose

$R_{Th\ adm} = (-\infty, v_{B0}] \times \{0,1\} \times \mathbb{R}$

and reduce the tests we have obtained, so far, to

$T_1 = R_{Th\ adm} \cap D_{block}$.

$T_2 = R_{Th\ adm} \cap D_{Red\text{-}Bo} = D_{Red\text{-}Bo}$

which yields the test set we proposed in section 2 (see Table 3.3 and Fig. 3.1).

	ΔV	gate	i
T_1			
	$(V_{Th}, V_{Bo}]$	{1}	[(res-Δ)*ΔV,(res+Δ)*ΔV]
T_2			
	$(V'_{Bo}, V_{Bo}]$	{0}	[0, δ]

Table 3.3 Set of two tests confirming the correct thyristor

Although in this example, the non-admissible range is related to the correct behavior, in general R_{adm} can also be chosen reflecting potential faults: for instance, if a pipe potentially has a crack, one might want to avoid high pressure even though this causes no problems for a proper pipe.

Lemma 3.2 does not prevent us from constructing observable tests that are not real, but rather an artificial result of the choice of model relations: a non-empty $D_i = p_{obs}(R_0) \setminus p_{obs}(R_i)$ may be due to choosing R_0 much larger than what is covered by the behavior, and D_i potentially contains only physically impossible values. In contrast, simply because nothing prevents us from causing inputs and observing observables, we have

Theorem 3.4

The existence of a deterministic input set ensures the existence of an observable and controllable test set in reality.

3.3 A Test Generation Algorithm

Here, we outline a family of algorithms (Fig. 3.3) based on Theorem 3.3, and discuss it briefly.

```
TI-SET = NIL
FOR R in MODEL-RELATIONS DO
(1) DI = R_adm ∩ p_cause(R_0) \ p'_cause(p_obs(R_0) ∩ p_obs(R))
(2) IF DI = ∅
      THEN "No (adm.) deterministic test input against" R
(3)   DI = R_adm ∩ p_cause(p_obs(R_0) \ p_obs(R))
        IF DI = ∅
        THEN "No (adm.) observable test against" R
             GOTO .NEXT
    Select TI ∈ TI-SET with DI ∩ TI ≠ ∅
    IF TI exists
(4)   THEN TI = TI ∩ DI
(5)   ELSE Append DI to TI-SET
    .NEXT
 END FOR
 FOR TI IN TI-SET
(6) Collect p_obs(R_0) ∩ p'^-1_cause(TI) in T-SET
```

Figure 3.3 An algorithm for generating (preferably deterministic) test inputs TI and test sets T confirming B_0

The algorithm iterates over the model relations of behaviors $B_i \neq B_0$ and attempts to create an admissible input set that discriminates between R_0 and R_i deterministically and in an observable way according to the above definition of DI_i (step 1). If this is impossible (2), it determines in (3) the admissible input set corresponding to an observable test (obtained as D_i according to Lemma 3.2) – which may fail, as well.

If there exist input sets from previous iterations with a non-empty intersection with the new DI, one of them is selected and replaced by this intersection (4). Thus, we

account for the behavior(s) corresponding to the current R without increasing the number of tests. Otherwise, the current DI is added as a new test input in itself (5). In step 6, an observable test set is constructed from the final input set according to Theorem 3.3. It is confirming B_0, if all R_i could be accounted for. The algorithm generates the two tests for the thyristor mentioned in section 2. The selection of TI for step 4 opens space for variations and heuristics. For instance, simply the first one with a non-empty intersection could be chosen, or the one with the largest intersection. The latter strategy always requires intersection with all existing input sets and assessment of the result, but may get closer to the optimum w.r.t. the number of tests generated. This algorithm produces the test set for the thyristor that is shown in Table 3.3.

If there exists a single test, the algorithm generates it in linear time. In other cases, it is quadratic w.r.t. the number of model relations (which may be less than the number of behaviors) and may fail to generate a test set of minimal cardinality. Its result, including whether or not an existing minimal cardinality test set is found, can depend on the ordering of the model relations. In many domains, it will pay off to use more elaborate and costly algorithms in order to reduce the number of tests required.

3.4 Making Test Generation Feasible through Model Abstraction

For physical systems with large or continuous domains and complex behavior, the question arises whether it is practically feasible to compute projections, intersections and set differences. The answer is that we do not have to. As in section 2, we want to make test generation for such domains feasible by performing it with model relations in an abstract representation (with small domains). We formalize this procedure and show its correctness. The key idea is simple: If $M(R_i)$ is a model of B_i, i.e.

$B_i \Rightarrow M(R_i)$,

and if R'_i is another relation (preferably in a finite domain) that specifies a weaker model, i.e.

$M(R_i) \Rightarrow M(R'_i)$,

then refuting $M(R'_i)$ suffices to rule out B_i. Hence, we can build test sets from such finite relations R'_i. The task is then to find conditions and a systematic way to generate models that are guaranteed to be weaker (in the logical sense specified above) by switching to a different representation ($\underline{v}'$,DOM'($\underline{v}'$)) with finite domains.

In (Struss 1992), a large class of transformations between representations is characterized by conditions that are both rather weak and natural:

Definition 3.5 (Representational Transformation)

A surjective mapping

τ: $DOM(\underline{v}) \rightarrow DOM'(\underline{v}')$

is a representational transformation iff

$\underline{v}(s) = \underline{v}_0 \quad \Rightarrow \quad \underline{v}'(s) = \tau(\underline{v}_0)$

$\underline{v}'(s) = \underline{v}'_0 \quad \Rightarrow \quad \exists \underline{v}_0 \in \tau^{-1}(\underline{v}'_0) \;\; \underline{v}(s) = \underline{v}_0.$

(See again Fig. 3.4). This simply means that, in the same situation, variables in the different representations have values related by τ. Under such representational transformations, models are preserved (Struss 1992):

Lemma 3.5

If

τ: $DOM(\underline{v}) \rightarrow DOM'(\underline{v}')$

is a representational transformation, then

$M(R) \Rightarrow M(\tau(R))$ *and* $M(R') \Rightarrow M(\tau^{-1}(R'))$.

This means, if we map a model relation from some original representation into a different one under a representational transformation the image will specify a weaker model, as required. In particular, we can choose a representation with a finite domain, construct (observable) confirming test sets and (deterministic) input sets in this representation from the transformed model relations and map them back to the original detailed representation.

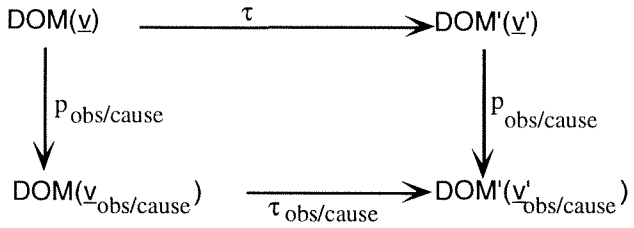

Figure 3.4 Diagram of projections and representational transformations

The following theorem states that this actually yields (deterministic) input sets and (observable) confirming test sets in the original representation, thus justifying the intuitive approach:

Theorem 3.6

Let

τ_{obs}: $DOM(\underline{v}_{obs}) \rightarrow DOM'(\underline{v}'_{obs})$

and

τ_{cause}: $DOM(\underline{v}_{cause}) \rightarrow DOM'(\underline{v}'_{cause})$

be representational transformations.

If $\{T'_i\}$ *is an observable confirming test set for* B_0 *then so is*

$\{T_i\} := \{\tau^{-1}_{obs}(T'_i)\}$.

If $\{TI_i\}$ *is a deterministic input set for* B_0, *then so is*

$\{TI_i\} := \{\tau^{-1}_{cause}(TI'_i)\}$.

In particular, qualitative domain abstraction (mapping real numbers to a set of landmarks and the intervals between them) is a representational representation. In the thyristor example, the landmarks can be chosen as 0, V_{Th}, V'_{Bo}, V_{Bo} for ΔV, and 0, $-\delta$, and δ for i. Ignoring the purely theoretical problem of separately treating the landmark points, this introduces quantity spaces Q_5 consisting of

neg := $(-\infty, 0]$
small := $(0, V_{Th}]$
medium := $(V_{Th}, V'_{Bo}]$
high := $(V'_{Bo}, V_{Bo}]$
too high := (V_{Bo}, ∞)

for ΔV and Q_3 with

\- := $(-\infty, -\delta)$
0 := $[-\delta, \delta]$
\+ := (δ, ∞)

for i, respectively. Mapping real values for ΔV and i to the intervals of Q_5 and Q_3, respectively, they are contained in, defines the qualitative abstraction

$$\tau_q : \mathbb{R} \times \{0,1\} \times \mathbb{R} \rightarrow Q_5 \times \{0,1\} \times Q_3$$

by

$$\tau_q((\Delta V_0, \ gate_0, \ i_0)) = (q_{v0}, \ gate_0, \ q_{i0}).$$

where

$$\Delta V_0 \in q_{v0} \in Q_5 \text{ and } i_0 \in q_{i0} \in Q_3$$

Under the reasonable assumption that the holding current is greater than the leakage current:

$$(res - \Delta) * V_{Th} > \delta ,$$

the representational transformation then induces model relations

$$R'_i := \tau_q(R_i) \subset Q_5 \times \{0,1\} \times Q_3$$

from the relations R_i in the real-valued representation. They are displayed in Table 3.4. (Since the two columns on the left for (ΔV, gate) represent the causes, one can immediately see the non-deterinism of the model $M(R'_{Red\text{-}Bo})$ for (medium, 0) which is due to the fact that it covers a whole class of behaviors with actual breakover voltage anywhere between V_{Th} and V'_{Bo}).

The table also shows the respective set differences D'_i in the new representation which can easily be determined from a comparison of the R'_{ok} column with respective R'_i column. The admissible range excludes *toohigh* from Q_5:

$$R'_{adm} = Q_5 \setminus \{\text{toohigh}\} \times \{0,1\} \times Q_3 ,$$

and the algorithm of section 3.3 intersects $D'_{Red\text{-}Bo}$ and D'_{punct} yielding the observable confirming test set

$T'_1 = \{\text{medium,high}\} \times \{1\} \times \{+\}\}$,

$T'_2 = \{(\text{high}, 0, 0)\}$

and the deterministic input set

$\{\{\text{medium,high}\} \times \{1\},$

$((\text{high},0)\}\}$.

Mapping back the tests to the real domain under τ_q^{-1} produces a test set

$\{\tau_q^{-1}(T'_1), \tau_q^{-1}(T'_2)\}$

$=\{ (V_{Th}, V_{Bo}] \times \{1\} \times (\delta, \infty),$

$(V'_{Bo}, V_{Bo}] \times \{0\} \times [-\delta, \delta]\}$

which covers the test set $\{T_1, T_2\}$ shown in Table 3.3 Of course, the abstract representation may be too coarse to allow for the separation of particular behaviors. We can use this as a criterion for selecting representations and behavior models, for instance, as the highest level that still allows to distinguish one behavior from the others.

One may be amazed that Theorem 3.6 requires τ_{obs} and τ_{cause} to be representational transformations, and tempted to find a weaker sufficient condition. This is briefly discussed in the following subsection which is not essential and may be skipped.

3.5 Preservation of Observable Distinctions

Recall the procedure for test generation: its basis is to identify *observable distinctions* of behavior model relations. The observability of this distinction must not be destroyed under back-transformation by τ^{-1}. In other words, if $\underline{v}'_1$ and $\underline{v}'_2$ can be observed as being distinct:

$$p'_{obs}(\underline{v}'_1) \neq p'_{obs}(\underline{v}'_2),$$

then the observable parts of their pre-images $\tau^{-1}(\underline{v}'_i)$ must also be disjoint. This seems to be the weakest sufficent condition we can impose, and it is captured by the following definition (which we formulate for the causes, as well, because this is needed for the construction of deterministic input sets for analogous reasons).

ΔV	**gate**	**i**						
		R'_{ok}	$R'_{Red\text{-}Bo}$	R'_{block}	R'_{punct}	$D'_{Red\text{-}Bo}$	D'_{block}	D'_{punct}
neg	0	0	0	0	-			0
neg	1	0	0	0	-			0
small	0	0	0	0	+			0
small	1	0	0	0	+			0
medium	0	0	0,1	0	+			0
medium	1	+	+	0	+		+	
high	0	0	+	0	+	0		0
high	1	+	+	0	+		+	
toohigh	0	+	+	0	+		+	
toohigh	1	+	+	0	+		+	

Table 3.4 The tuples constituting the relations R'_i and the set differences D'_i in the qualitative representation. The values for the current i, in the respective column complements the pair for (ΔV,gate) in each line.

Definition 3.6 (Faithful w.r.t. observable (causal) distinctions)

A representational transformation

τ: $DOM(v) \rightarrow DOM'(v')$

is called faithful w.r.t. observable distinctions iff

$\forall \underline{v}'_1, \underline{v}'_2 \in DOM'(\underline{v}')$ $(p'_{obs}(\underline{v}'_1) \neq p'_{obs}(\underline{v}'_2)$

$\Rightarrow p_{obs}(\tau^{-1}(\underline{v}'_1)) \cap p_{obs}(\tau^{-1}(\underline{v}'_2)) = \emptyset)$.

It is called faithful w.r.t. causal distinctions if the analogous property holds for p_{cause}.

The property that the restriction of τ to observables (causes, respectively) is also a representational transformation, which is used as a condition in the previous subsection, will be called decomposability:

Definition 3.7 (Decomposable)

If there exists a mapping

τ_{obs}: $DOM(\underline{v}_{obs}) \rightarrow DOM'(\underline{v}'_{obs})$

such that

$p'_{obs} \circ \tau = \tau_{obs} \circ p_{obs}$,

then τ *is called obs-decomposable.*

It is called cause-decomposable if there exists a

τ_{cause} : $DOM(\underline{v}_{cause}) \rightarrow DOM'(\underline{v}'_{cause})$

with

$p'_{cause} \circ \tau = \tau_{cause} \circ p_{cause}$.

(Here "$\circ$" is the composition of transformations). Note that we only require that τ_{obs} (τ_{cause} , resp.) be a *mapping*. This is because any such mapping "inherits" the properties of a representational transformation from the three involved mappings:

Lemma 3.7

If there exists τ_{obs} (τ_{cause}) *with the properties of Definition 3.7, then* τ_{obs} (τ_{cause}) *is a representational transformation.*

The following lemma states that this concept is only seemingly stronger than the one of Definition 3.6.

Lemma 3.8

A representational transformation

τ: $DOM(\underline{v}) \rightarrow DOM'(\underline{v}')$

is faithful w.r.t. observable (causal) distinctions iff τ *is obs-decomposable (cause-decomposable).*

Remark 3.9

τ_{cause} *being a representational transformation is also a* ***necessary*** *condition for the validity of backtransformation of tests in the following sense: If it is violated, we can construct behaviors and model relations such that there exist observable test sets with deterministic input sets for them in the abstract representation, but none in the stronger one. However, these constructed behaviors may be irrelevant to any real physical system, and the back-transformation of tests may work for the practical cases nevertheless.*

4 Testing Constituents in an Aggregate

Quite often the constituent to be tested is embedded in a particular context, namely an environment consisting of other interacting constituents, and only the entire aggregate can be controlled and observed. Our approach is general enough to cover this case.

We regard the aggregate as the constituent to be tested, and observables and causes are related to this aggregate constituent. The goal is to confirm one behavior of this aggregate constituent by refuting the other behaviors out of a certain set. This set is given as the behaviors of the aggregate resulting from the different behaviors of the constituent embedded in it.

More formally, let a constituent C_0 be in a particular context CTX consisting of constituents $C_1,...C_n$ with their respective variables. The aggregate is

$C_{agg}=\{C_j\}\cup\{C_0\}$,

and representations for describing the aggregate's behavior can be obtained from the representations for single constituents by taking the union of the local variables. For the sake of simplicity, we assume that all local relations are already specified in the aggregate representation. Issues that arise if the assumption is dropped are discussed in (Struss 1994).

If $M(R_j)$ are behavior models for constituents C_j, then

$R_{CTX}=\cap Rj$

specifies a corresponding behavior model for $CTX=\{C_j\}$. If $M(R_{io})$ are models of the behaviors B_i of C_0, then the relations

$R_i=R_{CTX}\cap R_{io}$

specify models of the behaviors of

$C_{agg}=CTX\cup\{C_0\}$

produced by the behaviors of C_0 in CTX. In applying the test generation algorithm to these relations, we can construct observable tests and deterministic test inputs for the behavior of C_{agg} that involves the particular behavior B_0 of C_0.

Corollary 4.1

Let

$C_{agg} = CTX \cup \{C_0\} = \{C_j\} \cup \{C_0\}$,

$R_{CTX} = \bigcap_j R_j$,

where the $M(R_j)$ *are behavior models of the* C_j.

Let $M(R_{i0})$ *be models of the behavior modes* B_i *of* C_0.

Define

$R_{agg\ i} := R_{CTX} \cap R_{i0}$,

$D_i := p_{obs}(R_{agg\ 0}) \setminus p_{obs}(R_{agg\ i})$

$DI_i := p_{cause}(R_{agg\ i}) \setminus$

$p'_{cause}(p_{obs}(R_{agg\ 0}) \cap p_{obs}(R_{agg\ i}))$.

If $M(R_{CTX})$ *holds, then each hitting set of sets* $\{T_k\}$ *of* $\{D_i\}$ *is an observable confirming test set for* B_0 *of* C_0, *and each hitting set of sets* $\{TI_k\}$ *of* $\{DI_i\}$ *is a deterministic input set for* B_0.

Since p_{cause} and p_{obs} project to input sets and observables of C_{agg}, the tests are observable and controllable *through* C_{agg}. Of course, this provides a confirming test set for B_0 of C_0, *only if* $M(R_{CTX})$ *holds.* This corresponds, for instance, to the widespread assumption that while testing a constituent, its context works properly. However, we can also generate tests based on the assumption that the context may contain particular faults, which, for instance, have been hypothesized by a diagnosis step.

By constructing all behavior modes of C_{agg} corressponding to a single fault of any constituent, we can generate a test set confirming the correctness of all constituents under this assumption.

5 Realization of Testing

Now we have to implement a test system, i.e. a program that takes the test inputs and the observed responses of the device and returns whether the respective behavior has been confirmed or refuted. For this purpose, we do not have to invent a new machinery but can apply an existing diagnostic system. Tests confirming a behavior are based on refuting models of all other behaviors. Refuting behaviors through observations is also the principle of consistency-based diagnosis (de Kleer, Mackworth & Reiter 1990), and we can implement testing through one of the consistency-based diagnosis engines, GDE$^+$ (Struss & Dressler 1989).

In more detail, GDE$^+$ represents a constituent by the set of behavior models $M(R_i)$. If a complete test set $\{T_k\}$ is observed, i.e. φ_V holds for some hitting set V of $\{T_k\}$, then we have

$$\forall T_k \ \exists s \in SIT \ \exists \underline{v}_k \in T_k \ \ \underline{v}(s)=\underline{v}_k.$$

By construction, there exists for each $D_i:=R_0\backslash R_i$ at least one $T_k \subseteq D_i$. Hence, it follows

$$\forall i\neq 0 \ \exists s \in SIT \ \exists \underline{v}_i \in D_i \ \ \underline{v}(s)=\underline{v}_i,$$

which means GDE$^+$ refutes all behaviors except B_0:

$$\forall i\neq 0 \ \exists s \in SIT \ \exists \underline{v}_i \notin R_i \ \ \underline{v}(s)=\underline{v}_i$$

$$\Rightarrow \forall i\neq 0 \ \neg M(R_i) \Rightarrow \forall i\neq 0 \ \neg B_i.$$

Then GDE$^+$ confirms B_0 by applying its "physical negation" rule (stating the completeness of the enumerated behaviors)

$$\neg B_1 \wedge \neg B_2 \wedge \ldots \wedge \neg B_n \Rightarrow B_0.$$

Of course, observation of a value outside R_0 lets GDE$^+$ refute B_0. In summary, GDE$^+$ makes the inferences required for the application of a deterministic input set.

Note that, for the purpose of testing, we can replace the constituent's model set $\{M(R_i)\}$ by the complements of the tests, $\{M(T^c_k)\}$, thus potentially reducing the number and, perhaps, the complexity of models to be checked. (Again, the details are discussed in (Struss 94)).

6 Discussion, Future and Related Work

6.1 What Is Achieved?

We make the rather strong claim that the theory presented here *solves* the problem of testing physical systems. It solves it "in principle", in the same sense as model-based diagnosis is a solution to the problem of fault localization and identification. By this, we want to emphasize two aspects:

- On the positive side, it is a *general* theory covering large classes of devices, for which there exists no formal theory or systematic solution of the testing problem today. All other solutions to test generation are only variations of this principled approach, perhaps by applying heuristics, making certain assumptions, or exploiting particularities of the domain (For instance, we can show that the D-algorithm (Roth 1980) is a specialization of our algorithm for digital circuit testing).
- On the problem side, it is a solution only "in principle", because it shifts the burden to the hard task of modeling. The application to a particular domain may require substantial work and even be impossible with the knowledge available about the domain. The crucial issues are in modeling and complexity. We briefly mention some of them.

6.2 Application Prerequisites and Problems

Knowledge about the Possible Faults: Particularly people from model-based diagnosis may be sceptical about the necessity of (complete sets of) fault models for this approach. However, knowledge (or assumptions) about the possible faults is not a drawback of our system, but is *inherent to the task of testing.* In constrast to diagnosis, where we may be content with refutation of (correct) behaviors, testing aims at confirming a particular behavior, usually the correct one. This is impossible, unless we make certain assumptions about the other possible behaviors, although this may happen unconsciously and implicitly. (This is why we are talking about testing of *physical systems,* and, for instance, not about testing system designs or software.) We may deliberately exclude some faults from the set of behaviors, or, more precisely, choose model relations that do not cover some fault behaviors, e.g. because we assume they are unlikely or irrelevant. For instance, we ignored thermal triggering of the thyristor and based the model on the assumption that the turn-ON time and spreading time are negligible. Our approach has the advantage to make such assumptions explicit (and the multiple modeling framework allows us to treat them as defeasible hypotheses, see (Struss 1992)).

Models of Behavior Modes: We have to be able to turn our knowledge about the correct and the faulty behavior into models, relational models in our case. The

representation through relations is quite natural for broad classes of physical systems. Note that the models are *not* required to be *deterministic* (remember the model of the class of thyristor faults called "Reduced V_{Bo}"). A major problem is finding appropriate models of devices with complex *dynamic* behavior. The thyristor, a dynamic device, illustrates that it can be possible to do the testing under temporal abstraction.

Abstract and Qualitative Models: Model abstraction is the key for the feasibility of the algorithm. But the models have to be strong enough to distinguish the behavior mode of interest from the other ones.

Structural Complexity: If aggregates are getting large, testing of constituents in its context may turn infeasible because of the number of fault models of the aggregate, (even though run time of several days for test generation may be insignificant compared to savings in testing time and costs). We do not expect the algorithm to handle systems with thousands of components in a flat structure. But first experiments suggest that it can produce results in a reasonable amount of time for devices which are complex enough to prohibit the completeness and/or optimality of manually generated tests. Currently, we are exploring binary-decision diagrams as a compact representation of the model relations. The core of the approach is finite constraint satisfaction. It will benefit from exploiting the specificity of the task, e.g. the device structure.

6.3 Perspectives

In this paper, we considered only testing with the goal of *confirming one particular mode.* Obviously, there is a generalization possible that creates tests for *identifying the present mode.* Testing for discrimination is relevant to diagnosis (Meerwijk & Preist 1992) and fits very well with a consistency-based diagnosis engine. It can be combined with probabilities of modes (and of tuples for non-deterministic models) and forms a generalization of the probe selection strategy used in (de Kleer & Williams 1987). Additionally, this approach provides a basis for a formal assessment of the (discriminating) power of representations. Testing in the context of diagnosis is the subject of another paper. Another direction is the exploitation of the strategy for design purposes: it allows to analyze whether or not and where it is (or would be) possible to detect, discriminate, and identify faults of constituents in a designed system, thus supporting design for testability and sensor placement (see (Chien, Doyle & Rouquette 1991), (Scarl 19991)).

In summary, we presented an approach to model-based test generation and testing that makes a large class of systems amenable to principled methods and well-founded algorithms. The exploitation of model abstraction is crucial to making the task practically feasible for an interesting class of technical systems, notwithstanding the fact that the general task of hypothesis testing is np-complete (McIlraith 1993). Finally, the basis of the theory is quite simple, simple enough to be powerful.

Acknowledgements

This work has been supported in part by the Christian-Doppler-Labor of the Technical University of Vienna.

References

Camurati, P., Medina, D., Prinetto, P., and Sonza, M.1990, *A Diagnostic Test Pattern Algorithm.* In: Proceedings IEEE International Test Conference, 52-58

Chien, S., Doyle, R., and Rouquette, N. 1991, *A Model-based Reasoning Approach to Sensor Placement for Diagnosability.* In: Working Notes of the Second International Workshop on Principles of Diagnosis, Milano, 181-190

de Kleer, J., Mackworth, A., and Reiter, R. 1990, *Characterizing Diagnoses.* In Proceedings of the AAAI 90, 324-330.

Gupta, A., and Welham, R.1989, *Functional Test Generation for Digital Circuits.* In: J. S. Gero (ed.), Proceedings of Artificial Intelligence in Engineering, Learning and Diagnosis, Elsevier

McIlraith, S. 1993, *Generating Tests Using Abduction.* In Working Papers of the Fourth International Workshop on Principles of Diagnosis, Aberystwyth, 223-235.

Meerwijk, A., and Preist, C.1992, *Using Multiple Tests for Model-based Diagnosis.* Working Papers of the Third International Workshop on Principles of Diagnosis, Rosario

Roth, G. P. 1980, *Computer Logic, Testing, and Verification.* Rockville: Computer Science Press.

Scarl, E. 1991, *Analysis of Diagnosability.* In: Working Notes of the Second International Workshop on Principles of Diagnosis, Milano, 191-200

Struss, P. 1992, *What's in SD? Towards a Theory of Modeling for Diagnosis.* In: Hamscher, W. Console, L., and de Kleer, J. eds., Readings in Model-based Diagnosis. San Mateo: Morgan Kaufmann: 419-449.

Struss, P. 1994, *A Theory of Testing Physical Systems Based on First Principles*, Technical Report 94/63, Christian-Doppler-Labor, Technical University of Vienna.

Struss, P. 1994a, *Multiple Models of Physical Systems - Modeling Intermittent Faults, Inaccuracy and Tests in Diagnosis.* To appear in: Annals of Mathematics and Artificial Intelligence .

Struss, P., Dressler, O. 1989, *"Physical Negation" - Integrating Fault Models into the General Diagnostic Engine.* In Proc. 11th Int. Joint Conf. on Artificial Inteligence, Detroit, MI, 1318-1323.