Dynamic domain abstraction through meta-diagnosis

Johan de Kleer

Palo Alto Research Center,
3333 Coyote Hill Road, Palo Alto, CA 94304 USA
dekleer@parc.com

Abstract. One of the most powerful tools designers have at their disposal is abstraction. By abstracting from the detailed properties of a system, the complexity of the overall design task becomes manageable. Unfortunately, faults in a system need not obey the neat abstraction levels of the designer. This paper presents an approach for identifying the abstraction level which is as simple as possible yet sufficient to address the task at hand. The approach chooses the desired abstraction level through applying model-based diagnosis at the meta-level, i.e., to the abstraction assumptions themselves.

Key words: Abstraction, diagnosis, qualitative reasoning, model-based reasoning.

1 Introduction

Of the many tools designers have at their disposal, abstraction is one of the most powerful. By abstracting from the detailed properties of a system, the complexity of the overall design task becomes manageable. For example, a computer engineer can focus on the logic level without concern for the properties of the individual transistors which make up a particular gate, and a chip designer can layout a chip without being concerned with the fabrication steps needed to construct it. Abstraction allow designers to partition concerns into independent black boxes and is one of the most important ideas underlying the design of modern technology.

Unfortunately, faults in a system need not obey the neat abstraction levels of the designer. A fault in a few transistors can cause an Intel Pentium processor to generate an ocasional incorrect floating point result. To understand this fault requires transcending the many abstraction levels between software and hardware. A PC designer can focus on functional layout without being concerned about the physical layout and its thermal properties. However, a technician must determine that the processor crashed because dust sucked into the processor fan clogged the heatsink and allowed the processor temperature to rise to such a dangerous level that the PC automatically shut down. As a consequence diagnostic reasoning is inherently messy and complex, as it involves crossing abstraction boundaries never contemplated by the designers.

Existing model-based reasoning has addressed a number of types of abstraction.

- Range abstraction. The ranges of variables are abstracted, e.g., instead of a continuous quanity it might be represented by the qualitative values of $-$, 0 or $+$. [1–4]
- Structural abstraction. Groups of components are abstracted to form hierarchies [5, 6].
- Model selection. Approaches to choosing among a collection of hand-constructed models [7, 8]

In this paper we present a new type of abstraction (*domain* abstraction): changing the physical principles which underlie models, such as moving from the 0/1 level to currents and voltages and providing a systematic approach to choosing the appropriate domain for the diagnostic task.

Choosing the right domain abstraction level requires balancing two opposing desiderata. Reasoning at the highest abstraction level is the simplest. Unfortunately, it may be inadequate to analyze or troubleshoot the system. Instead, the system needs to be analyzed at a more detailed level. On the other hand, reasoning at too low of a level can require enormously more computational resources and difficult to obtain parameters, and it generates more complicated analyses. As Albert Einstein reportedly said: "Make everything as simple as possible, but not simpler."

Technicians expect that systems are non-intermittent and that the schematic is an accurate description of the physical system. Consider the simple analog circuit of Fig. 1. Suppose a technician measures the current to be 0 ampere (1 is expected), which leads to an inference that the resistor is faulty, but repeating the measurement gives 1 ampere. Either the resistor is intermittent or there is a fault in the connections. The technician must change abstraction level to diagnose this system further by, for example, checking the connections or further tests on resistor R itself.

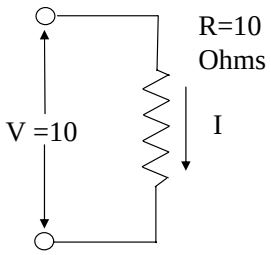

Fig. 1. Simple analog diagnosis problem.

Consider a circuit of three logic inverters in sequence, with its output fedback to its input (Fig. 2). At the usual gate level of analysis, an inverter simply

complements its input. This circuit has no inputs, so we need to consider the possible values at the connecting nodes. Assume the input to inverter A is 0. Its output must be 1. The output of B must be 0. The output of D must be 1. This is impossible, as we assumed it was 0. Conversely, assume the input to inverter A is 1. Its output must be 0. The output of B must be 1. The output of D must be 0. This is impossible, as we assumed it was 1. Therefore, the input of inverter A can neither be 0 or 1. Also, the inputs of inverters B or D cannot be 0 or 1. Somehow the circuit is contradictory when modeled as logic gates. Thus, one of the components A, B or D must be faulted. Suppose the technician chooses to systematically remove and check each of these three components for proper functioning. If each component is confirmed to be correct, the technician has encountered an impasse.

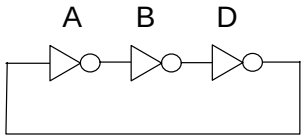

Fig. 2. A very simple circuit which yields a contradiction when analyzed as combinational logic; yet its a perfectly reasonable fault-free circuit with well-defined behavior.

Analyses that result in contradictions are the most important indicator that the level of abstraction used is too simplistic. In this paper we present a general reasoner which automatically descends abstraction layers to perform needed analyses, and which does not descend abstraction levels needlessly. This approach is broadly applicable, but we explore these ideas in the context of digital circuits with messier models that include failures in connections, intermittents, and temporal behaviors.

2 Meta-Diagnosis

Fig. 3 characterizes the basic architecture of a typical model-based, component-based diagnosis engine. Given the component topology (e.g., the schematic for analog circuits), component models (e.g., resistors obey ohm's law) and observations (e.g., the voltage across resistor $R6$ is 4 volts), it computes diagnoses which explain all the observations. Observations inconsistent with expectations guide the discovery of diagnoses. When the MBD engine can find no diagnoses it signals a failure.

Suppose we need to troubleshoot the circuit of Fig. 2. Most diagnosis systems would immediately conclude that some subset of the components $\{A, B, D\}$ is faulted. However, testing each inverter individually demonstrates that all the components are good. As a consequence, most algorithms would report an unresolvable contradiction.

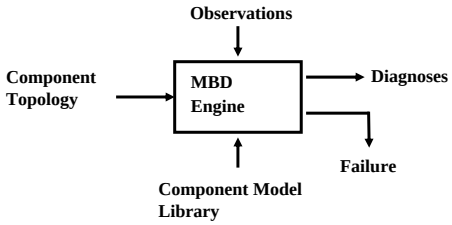

Fig. 3. Basic architecture of a model-based diagnosis engine.

The architecture of Fig. 4 includes two model-based diagnosis engines. The top model is used to identify the best abstraction level, and the bottom model performs the actual system diagnosis. This composite architecture has the same inputs as the basic architecture with one additional input: the abstraction library. The 'Applicable Models' module identifies all the applicable abstractions for the component topology. The 'Modeler' module uses the preferred meta-diagnosis to construct conventional model-based diagnosis models using the 'Component Model Library.'

Consider the example of Fig. 2. The component topology is simply the circuit schematic as before. The system observations are as before (e.g., the output of A is 1). The component model library contains different models for gate behavior (e.g., boolean, analog, thermal, temporal, etc.). The new input, the abstraction library, is the set of all possible abstractions. Instead of a usual component topology, the abstraction MBD engine is provided with a set of possible abstractions applicable to the given system to be diagnosed. Initially, there are no meta-observations, so the preferred diagnosis is the one at the most abstract level (analogous to all components working). Therefore, the domain MBD engine will perform diagnosis in the usual way with the most abstract models. Suppose each gate is physically checked, leading to the observations that A, B and D are working correctly. The domain model-based engine now fails as it has found an unresolvable contradiction. This invokes the abstraction MBD engine as an observation. As analysis proceeds, the preferred meta-diagnosis will descend abstraction levels. For the purposes of this paper, the preferred meta-diagnosis is one minimal cardinality meta-diagnosis.

3 Formalization

This section summarizes the formal framework for model-based diagnosis we use in the rest of the paper [9, 10]. In order to distinguish between domain and abstraction AB literals, we state the usual framework in terms of domain AB_d literals.

Definition 1. *A system is a triple (SD,COMPS, OBS) where:*

1. *SD, the system description, is a set of first-order sentences.*

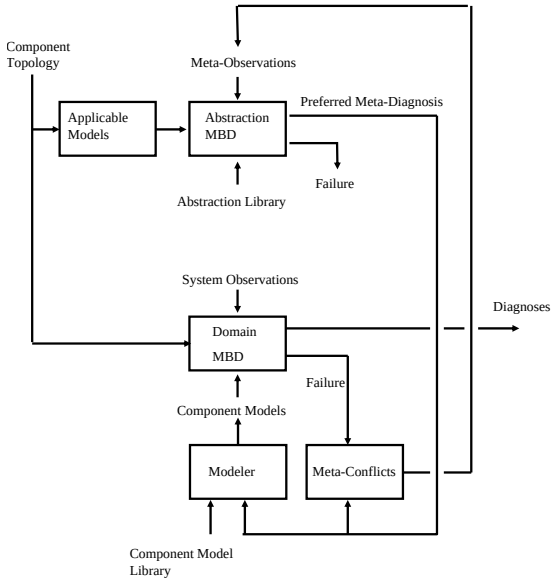

Fig. 4. Architecture of an abstraction-based, model-based diagnosis engine.

2. *COMPS, the system components, is a finite set of constants.*
3. *OBS, a set of observations, is a set of first-order sentences.*

In Fig. 3 *SD* is the component topology and component model library, and *COMPS* is the set of components in the component topology.

Definition 2. *Given two sets of components Cp and Cn define* $\mathcal{D}_d(Cp, Cn)$ *to be the conjunction:*

$$\Big[\bigwedge_{c \in Cp} AB_d(c)\Big] \wedge \Big[\bigwedge_{c \in Cn} \neg AB_d(c)\Big].$$

Where $AB_d(x)$ represents that the component x is ABnormal (faulted).

A diagnosis is a sentence describing one possible state of the system, where this state is an assignment of the status normal or abnormal to each system component.

Definition 3. *Let* $\Delta \subseteq COMPS$. *A diagnosis for (SD,COMPS,OBS) is* $\mathcal{D}_d(\Delta, COMPS - \Delta)$ *such that the following is satisfiable:*

$$SD \cup OBS \cup \{\mathcal{D}_d(\Delta, COMPS - \Delta)\}$$

Definition 4. *An* AB_d*-literal is* $AB_d(c)$ *or* $\neg AB_d(c)$ *for some* $c \in COMPS$.

Definition 5. *An* AB_d*-clause is a disjunction of* AB_d*-literals containing no complementary pair of* AB_d*-literals.*

Definition 6. *A conflict of (SD,COMPS,OBS) is an AB_d-clause entailed by $SD \cup OBS$.*

3.1 Formalizing Abstraction

The abstraction MBD is defined analogously:

Definition 7. *An abstraction system is a triple (SD,ABS, OBS) where:*

1. *SD, constraints among possible abstractions, is a set of first-order sentences.*
2. *ABS, the applicable abstractions, is a finite set of constants.*
3. *OBS, a set of meta-observations, is a set of first-order sentences.*

Definition 8. *Given two sets of abstractions Cp and Cn define $\mathcal{D}_a(Cp, Cn)$ to be the conjunction:*

$$\Big[\bigwedge_{c \in Cp} AB_a(c)\Big] \wedge \Big[\bigwedge_{c \in Cn} \neg AB_a(c)\Big].$$

Where $AB_a(x)$ represents that the abstraction x is ABnormal (cannot be used).

A meta-diagnosis is a sentence describing one possible state of the system, where this state is an assignment of the status normal or abnormal to each system component.

Definition 9. *Let $\Delta \subseteq ABS$. A meta-diagnosis for (SD,ABS,OBS) is $\mathcal{D}_a(\Delta, ABS - \Delta)$ such that the following is satisfiable:*

$$SD \cup OBS \cup \{\mathcal{D}_a(\Delta, ABS - \Delta)\}$$

Definition 10. *An AB_a-literal is $AB_a(c)$ or $\neg AB_a(c)$ for some $c \in ABS$.*

Definition 11. *An AB_a-clause is a disjunction of AB_a-literals containing no complementary pair of AB_a-literals.*

Definition 12. *A meta-conflict of (SD,ABS,OBS) is an AB_a-clause entailed by $SD \cup OBS$.*

4 Example of a Lattice of Models

To illustrate these ideas we use 3 axes of abstraction:

- Model of connections as in [11] which is an improvement over [12, 13].
- Model of non-intermittency [14] or intermittency [15]
- Model of time [16].

The corresponding AB_a literals are:

- $\neg AB_a(C)$ represents the abstraction that connections need not be modeled.
- $\neg AB_a(I)$ represents the abstraction that the system is non-intermittent.

– $\neg AB_a(T)$ represents the abstraction that temporal behavior need not be modeled.

Fig. 5(a) shows a portion of the abstraction space for digital circuits along three of the axes of abstraction. This lattice is identical in structure to the ones used in conventional model-based diagnosis for system diagnoses. In conventional model-based diagnosis, each node represents a candidate diagnosis which explains the observations. Each node in Fig. 5(b) represents a candidate meta-diagnosis. The bottom node in the figure represents the meta-diagnosis in which connections, time, and intermittency are not relevant: $\neg AB_a(T) \wedge \neg AB_a(C) \wedge \neg AB_a(I)$. Under the principle that we want to find the simplest meta-diagnosis which explains the observations (and no simpler), we are primarily interested in the minimal diagnoses. For brevity sake, we name meta-diagnoses with the letters corresponding to the abstractions which are AB_a. For example, the meta-diagnosis $\neg AB_a(T) \wedge \neg AB_a(C) \wedge AB_a(I)$ is named by I.

For the example in Fig. 2, analysis immediately detects a contradiction and the meta-conflict: $AB_a(T) \vee AB_a(C)$. (This contradiction cannot depend on $AB_a(I)$ as there is only one observation time so far). Fig. 5b illustrates the resulting meta-diagnosis lattice. Every meta-diagnosis below the curve is eliminated. The minimal meta-diagnoses are T and C.

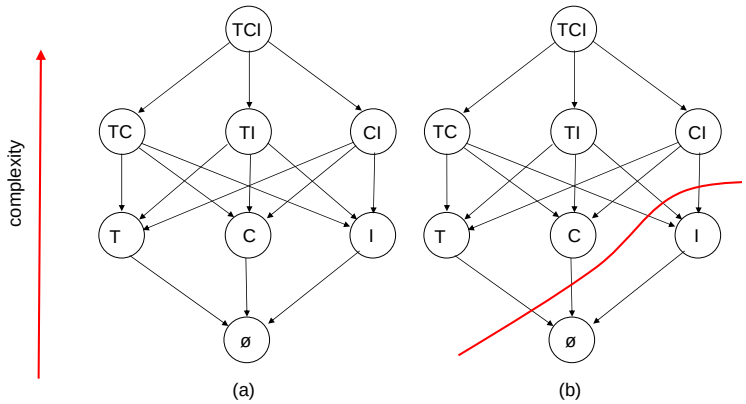

Fig. 5. (a) Meta-Diagnosis lattices for digital gates. T indicates temporal models; C indicates connection models; I indicates intermittent models. (b) The meta-conflict $AB_a(T) \vee AB_a(C)$ rules out all meta-diagnoses below the curved line. The minimal meta-diagnoses are T and C.

5 Modeling Components

The conventional MBD model for an inverter is (presuming the usual background axioms define the appropriate functions, domains, and ranges):

$$\text{INVERTER}(x) \rightarrow \Big[\neg AB_d(x) \rightarrow [in(x,t) = 0 \equiv out(x,t) = 1]\Big].$$

As this model presumes connections and temporal behavior need not be modeled, in our new framework it is written as:

$$\neg AB_a(T) \wedge \neg AB_a(C) \rightarrow \Big[\text{INVERTER}(x) \rightarrow \Big[\neg AB_d(x) \rightarrow [in(x,t) = 0 \equiv out(x,t) = 1]\Big]\Big].$$

When modeling an inverter as having a delay Δ, the model changes to (labeled T in Fig. 5):

$$AB_a(T) \wedge \neg AB_a(C) \rightarrow \Big[\text{INVERTER}(x) \rightarrow \Big[\neg AB_d(x) \rightarrow [in(x,t) = 0 \equiv out(x,t+\Delta) = 1]\Big]\Big].$$

5.1 Connection Models

To model the inverter to accommodate faults in connections, including bridge faults, requires the introduction of new formalisms. What follows is a brief summary of the formalism presented in [11]. Each terminal of a component is modeled with two variables, one which models how the component is attempting to influence its output (roughly analogous to current), and the other which characterizes the result (roughly analogous to voltage). There are 5, mutually inconsistent, qualitative values for the influence of a component on a node (we refer to these as "drivers").

- $d(-\infty)$ indicates a direct short to ground.
- $d(0)$ pull towards ground (i.e., 0).
- $d(R)$ presents a high (i.e., draws little current) passive resistive load.
- $d(1)$ pull towards power (i.e., 1).
- $d(+\infty)$ indicates a direct short to power.

There are three possible qualitative values for the resulting signal:

- $s(0)$ the result is close enough to ground to be sensed as a digital 0.
- $s(x)$ the result is neither a 0 or 1.
- $s(1)$ the result is close enough to power to be sensed as a digital 1.

Using this formalism produces considerably more detailed component models. We need to expand the $A \equiv B$ in the inverter model to $(A \rightarrow B) \wedge (B \rightarrow A)$. The left half of the inverter model is:

$$\neg AB_a(T) \wedge AB_a(C) \rightarrow \Big[\text{INVERTER}(x) \rightarrow \Big[\neg AB_d(x) \rightarrow \\ \big[[s(in(x,t)) = s(0) \rightarrow d(out(x,t)) = d(1)] \\ \wedge[s(in(x,t)) = s(1) \rightarrow d(out(x,t)) = d(0)] \\ \wedge d(in(x,t)) = d(R) \wedge [d(out(x,t)) = d(0) \vee d(out(x,t)) = d(1)]\big]\Big]\Big].$$

We need explicit models to describe how the digital signal at a the node is determined from its drivers. Let $R(v)$ be the resulting signal at node v and $S(v)$ be the collection of drivers of node v. Intuitively, the model for a node is:

- If $d(-\infty) \in S(v)$, then $R(v) = s(0)$.
- If $d(+\infty) \in S(v)$, then $R(v) = s(1)$.
- If $d(0) \in S(v)$, then $R(v) = s(0)$.
- Else, if all drivers are known, and the preceding 3 rules do not apply, then $R(v) = s(1)$.

The resulting model for the node x will depend on $\neg AB_d(x)$ and $AB_a(C)$.

Modeling the inverter to more accurately describe both temporal and causal behavior (labeled TC in Fig. 5):

$$AB_a(T) \wedge AB_a(C) \rightarrow \Big[\text{INVERTER}(x) \rightarrow \Big[\neg AB_d(x) \rightarrow \\ \big[[s(in(x,t)) = s(0) \rightarrow d(out(x,t+\Delta)) = d(1)] \\ \wedge[s(in(x,t)) = s(1) \rightarrow d(out(x,t+\Delta)) = d(0)] \\ \wedge d(in(x,t)) = d(R) \wedge [d(out(x,t+\Delta)) = d(0) \vee d(out(x,t+\Delta)) = d(1)]\big]\Big]\Big].$$

The connection models also allow arbitrary bridge faults among circuit nodes. These are described in much more detail in [11].

5.2 Modeling Non-Intermittency

Fig. 6 shows an example where assuming non-intermittency improves diagnostic discrimination. The circuit's inputs and outputs are marked with values observed at times: T_1 and T_2. Note that at T_1, the circuit outputs a correct value and that at T_2, the circuit outputs an incorrect one. By assuming the Or gate behaves non-intermittently, we can establish that the Xor gate is faulty as follows:

If Xor is good, then $\text{In}_1(\text{Xor}, T_1) = 1$. This follows from $\text{In}_2(\text{Xor}, T_1) = 0$, $\text{Out}(\text{Xor}, T_1) = 1$ and the behavior of Xor. Similarly, if Xor is good, then $\text{In}_1(\text{Xor}) = 0$ at T_2. However, if Or behaves non-intermittently, then $\text{In}_1(\text{Xor}, T_2) = 1$. This follows because Or has the same inputs at both T_1 and T_2 and must produce the same output. Thus we have two contradictory predictions for the value of $\text{In}_1(\text{Xor}, T_2)$. Either Xor is faulty or Or is behaving intermittently. Assuming non-intermittency means Xor is faulty.

All the inferences follow from the defining of non-intermittency:

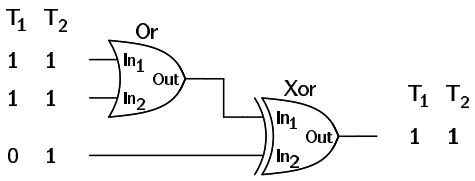

Fig. 6. The power of non-intermittency.

Definition 13. *[14] A component behaves non-intermittently if its outputs are a function of its inputs.*

This definition sanctions the following inference: if an input vector $\overline{X}$ is applied to an nonintermittent component at time T, and output Z is observed, then in any other observation T', if $\overline{X}$ is supplied as input, Z will be observed as output.

For the Or-Xor example, the axioms added are:

$$\forall t.\mathrm{Out}(\mathrm{Or},t) = F(\mathrm{Or},\mathrm{In}_1(\mathrm{Or},t),\mathrm{In}_2(\mathrm{Or},t))$$

$$\forall t.\mathrm{Out}(\mathrm{Xor},t) = F(\mathrm{Xor},\mathrm{In}_1(\mathrm{Xor},t),\mathrm{In}_2(\mathrm{Xor},t))$$

F is a single fixed function for all non-intermittency axioms.

These axioms are implemented in the ATMS/HTMS-based reasoner by deriving prime implicates as follows. At time T_1:

$$AB_d(\mathrm{Xor}) \vee [F(\mathrm{Or},1,1) = 1].$$

At time T_2:

$$AB_d(\mathrm{Xor}) \vee [F(\mathrm{Or},1,1) = 0].$$

Which combine to yield $AB_d(\mathrm{Xor})$.

In the intermittent case, the observation at T_1 equally weights Xor and Or as being correct. If there were other components in the system not affected by the measurement, the observation at T_1 lowers the posterior fault probabilities of Xor and Or.

5.3 Automatic Generation of Models

The more detailed component models can usually be generated automatically from the most abstract models in a systematic way. In our implementation, the T, C, and I models are automatically derived from the basic $\emptyset$ models by a set of modeling schemas. Consider the most abstract model of an inverter:

$$\mathrm{INVERTER}(x) \rightarrow \Big[\neg AB_d(x) \rightarrow [in(x,t) = 0 \equiv out(x,t) = 1]\Big].$$

We use the convention that the function *in* refers to inputs, and the function *out* refers to outputs. A non-temporal model can be converted to a simple gate-delay model by replacing every occurrence of $out(x,t)$ (or $out_j(x,t)$) with $out(x,t+\Delta)$.

A non-connection model can be converted to a connection one by first expanding implications, replacing all $in(x,t) = y$ with $s(in(x,t)) = s(y)$ and $d(in(x,t)) = d(R)$ and replacing all $out(x,t) = y$ with $d(out(x,t)) = d(y)$, and adding the usual domain axioms for new variable values.

Non-intermittency requires no change to the component models themselves, but the axioms of Section 5.2 need to be added to the models supplied to the domain MBD.

6 The Meta-Diagnosis Loop

6.1 $\emptyset \rightarrow T$

Consider the three inverter example of Fig. 2. The most abstract meta-diagnosis is:

$$\neg AB_a(T) \wedge \neg AB_a(C) \wedge \neg AB_a(I).$$

This meta-diagnosis is supplied to the 'Modeler' module for Fig. 4 which chooses the component models at the meta-diagnosis level. The models for all three inverters are given by the TC model of Section 5. This produces a failure because all components are known to be good. The 'Meta-Conflicts' module of Fig. 4 will construct the meta-conflict:

$$AB_a(T) \vee AB_a(C).$$

$AB_a(I)$ is trivially excluded from the meta-conflict because non-intermittency inferences can only arise when the system has been observed at multiple times.

The abstraction MBD identifies two minimal meta-diagnoses T and C. If both are equally likely, it arbitrarily picks one. Suppose C is chosen. The C models do not resolve the inconsistency either. Fig. 7 illustrates the following sequence of inferences with the connection models: (1) Assume the input of A is 1, (2) the causal inverter model drives its output down towards 0, (3) the input of gate B presents a high resistance (low-current) load to its node, (4) the connection model sets the node to 0, (5) the inverter model on B drives its output towards 1, (6) gate D presents a high resistance (low current) load, (7) the connection model sets its node to 1, (8) the inverter drives its output to 0, (9) gate A presents a low resistance (low current) load, and (10) the node model sets the node to 0 which contradicts the input of A being 1. An analogous analysis for the input of A being 0, yields a contradiction as well. The only remaining possible cardinality one diagnosis is T.

Using the temporal T models for the inverters produces a consistent analysis illustrated in Table 1. This circuit is the familiar ring oscillator [17].

6.2 $\emptyset \rightarrow I$

Consider the Or − Xor circuit again (Fig. 6). For clarity assume the circuit has one fault. As derived in the Section 5.2, under the $\emptyset$ models, Xor must be faulted.

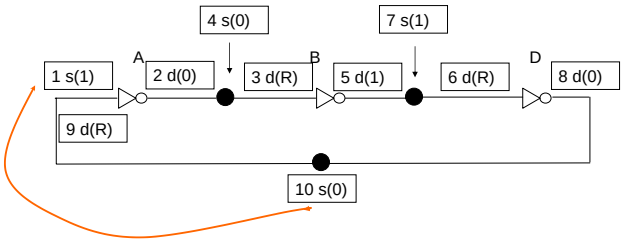

Fig. 7. Modeling connections does not remove the failure.

Table 1. Outputs of the inverters of a ring oscillator after t gate delays. The oscillator takes 6 gate delays to return to its initial state, thus the output is a square wave with a period of 6 times the gate delay.

t	0	1	2	3	4	5	6
A	1	1	1	0	0	0	1
B	1	0	0	0	1	1	1
C	0	0	1	1	1	0	0

Suppose we measure the output of the Or gate at T_1 and T_2 to be 1 and then 0 respectively. In this case, we have derived the meta-conflict:

$$AB_a(T) \vee AB_a(C) \vee AB_a(I).$$

There are now three minimal candidate meta-diagnoses: T,C,I. The T meta-diagnosis immediately results in a failure yielding the meta-conflict:

$$AB_a(C) \vee AB_a(I).$$

The meta-diagnosis I yields a consistent point of view: Or is failing intermittently. The C meta-diagnosis cannot explain the observation:

$$AB_a(T) \vee AB_a(I).$$

6.3 $\emptyset \rightarrow C$

Consider the Or−Xor example again before the output of the Or gate is observed. Again, for clarity assume the circuit has one fault. Suppose Xor is replaced and the same symptoms reoccur. In this case, both the C and I meta-diagnoses are consistent. Under the I meta-diagnosis, the circuit contains two possible faults:

- Xor is faulted.
- Or is faulted.

The C meta-diagnosis is consistent, with 3 possible faults:

- The node at the output of Xor is shorted to power.
- The connection from the output Xor gate to the node is open and thus it floats to 1.
- The connection to in_2(Xor) is shorted to ground.

6.4 $\emptyset \rightarrow TCI$

Tasks which require a TCI preferred meta-diagnosis are complicated, but they do occur. Consider the four inverter system of Fig. 8. The input to inverter

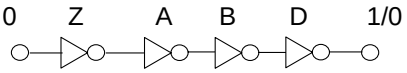

Fig. 8. A very simple circuit containing a very hard to pinpoint fault.

Z is held constant at 0. We assume single faults. Observing the output D is usually 0, but outputs 1s with no pattern. The observation $D = 1$ indicates that one of Z, A, B, D is faulted. However, a subsequent observation of $D = 0$ is inconsistent yielding the meta-conflict: $AB_a(T) \vee AB_a(C) \vee AB_a(I)$. No fault in the connections can produce the observations, therefore: $AB_a(T) \vee AB_a(I)$. No temporal fault can lead to this behavior either, so: $AB_a(I)$. Under meta-diagnosis I, the output of A is measured — it is usually 0, but sometimes 1. The output of Z is measured — it is usually 1, but sometimes 0. Therefore Z must be intermittently faulted (under meta-diagnosis I), but replacing it does not change the symptoms. This yields the meta-conflict: $AB_a(T) \vee AB_a(C)$. The CI meta-diagnosis also leads to an inconsistency. There is no fault within the connections that can explain the observations. Likewise there is no fault within the TI meta-diagnosis. The only meta-diagnosis that can explain the symptoms is TCI. The actual fault is an intermittent short between the output of D and output of Z. As the input to Z is 0, its output is 1. The connection models for digital gates are 0-dominant, so that, if a 0 from the output of D were fedback through an intermittent short, it would drive the input to A to 0. Thus for those times in which the intermittent short was manifest, the circuit would be a ring oscillator.

7 Implementation

The analyses described in this paper have been implemented within the ATMS/HTMS framework [9, 18]. Each domain or abstraction literal is represented by an explicit ATMS assumption in one ATMS instance. A fuller description of the T, C, and I abstractions can be found in [14, 11, 15, 16]. The ATMS/HTMS architecture provides a unified framework to reason over any assumptions, be they about components or abstractions.

8 Related Work

Automated model abstraction has a long tradition in Artificial Intelligence. The graph of models [7] is similar to the meta-diagnosis lattice (Fig. 5) and ana-

lyzes conflicts to identify which modeling parameters to change. It is focused on design and analysis and the models that are constructed by hand. It does not use diagnosis to guide the search for models, nor is it applied to diagnosis in some domain. Work on compositional modeling [8] also uses ATMS assumptions to represent domain abstractions and conflicts to guide the search for models. Again, the models are constructed by hand and do not use diagnosis at the domain or meta-levels. In context-dependent modeling [19] there is typically a much larger space of model fragments to choose from and explicit context information is used to guide the selection of the domain models. The task is to construct the best causal explanation for a physical phenomena. Yet again, the models are constructed by hand and do not use diagnosis at the domain or meta-levels.

In the model-based diagnosis literature, there has been considerable work on diagnostic assumptions and selecting appropriate models for a diagnostic task [1, 2]. This paper focuses primarily on assumptions associated with choosing domain abstractions.

There has been considerable research on structural abstraction [5, 6] where groups of components are combined to form larger systems to reduce computational complexity. [3] describes how the task can be used to partition the value of a variable into the qualitative values needed to solve a task. [4] presents another approach to partition the value of a variable into qualitative ranges to reduce complexity when there is limited observability of the variables.

9 Conclusions

This paper has presented a general approach to selecting the best domain abstraction level to address a task and has demonstrated it within the context of digital gates. In the case of digital gates the component models can be automatically generated from the basic models using domain schemas.

Acknowledgments. Daniel G. Bobrow and Elisabeth de Kleer provided many useful comments.

References

1. Struss, P.: What's in SD? Towards a theory of modeling for diagnosis. In Console, L., ed.: Working Notes of the 2st Int. Workshop on Principles of Diagnosis. Technical Report RT/DI/91-10-7, Dipartimento di Informatica, Universitá di Torino, Torino, Italy (October 1991) 41–51
2. Struss, P.: A theory of model simplification and abstraction for diagnosis. In: Proc. 5th Int. Workshop on Qualitative Physics, Austin, TX (May 1991)
3. Sachenbacher, M., Struss, P.: Task-dependent qualitative domain abstraction. Artif. Intell. **162**(1-2) (2005) 121–143
4. Torta, G., Torasso, P.: Automatic abstraction in component-based diagnosis driven by system observability. In Gottlob, G., Walsh, T., eds.: IJCAI, Morgan Kaufmann (2003) 394–402

5. Chittaro, L., Ranon, R.: Hierarchical model-based diagnosis based on structural abstraction. Artif. Intell. **155**(1-2) (2004) 147–182
6. Hamscher, W.C.: XDE: Diagnosing devices with hierarchic structure and known component failure modes. In: Proc. 6th IEEE Conf. on A.I. Applications, Santa Barbara, CA (March 1990) 48–54
7. Addanki, S., Cremonini, R., Penberthy, J.S.: Reasoning about assumptions in graphs of models. In: Proc. 11th Int. Joint Conf. on Artificial Intelligence, Detroit, MI (August 1989) 1432–1438
8. Falkenhainer, B., Forbus, K.D.: Compositional modeling: Finding the right model for the job. Artificial Intelligence **51**(1-3) (1991) 95–143 Also in: J. de Kleer and B. Williams (eds.) *Qualitative Reasoning about Physical Systems II* (North-Holland, Amsterdam, 1991 / MIT Press, Cambridge, Mass., 1992).
9. de Kleer, J., Williams, B.C.: Diagnosing multiple faults. Artificial Intelligence **32**(1) (April 1987) 97–130 Also in: *Readings in NonMonotonic Reasoning*, edited by Matthew L. Ginsberg, (Morgan Kaufmann, 1987), 280–297.
10. de Kleer, J., Mackworth, A., Reiter, R.: Characterizing diagnoses and systems. Artificial Intelligence **56**(2-3) (1992) 197–222
11. de Kleer, J.: Modeling when connections are the problem. In: Proc 20th IJCAI, Hyderabad, India (2007) 311–317
12. Böttcher, C.: No faults in structure? how to diagnose hidden interactions. In: IJCAI. (1995) 1728–1735
13. Böttcher, C., Dague, P., Taillibert, P.: Hidden interactions in analog circuits. In Abu-Hakima, S., ed.: Working Papers of the Seventh International Workshop on Principles of Diagnosis, Val Morin, Quebec, Canada (October 1996) 36–43
14. Raiman, O., de Kleer, J., Saraswat, V., Shirley, M.H.: Characterizing non-intermittent faults. In: Proc. 9th National Conf. on Artificial Intelligence, Anaheim, CA (July 1991) 849–854
15. de Kleer, J.: Diagnosing intermittent faults. In: 18th International Workshop on Principles of Diagnosis, Nashville, USA (2007) 45–51
16. de Kleer, J.: Troubleshooting temporal behavior in "combinational" circuits. In: 18th International Workshop on Principles of Diagnosis, Nashville, USA (2007) 52–58
17. Wikipedia: Ring oscillator — Wikipedia, the free encyclopedia (2007) [Online; accessed 12-February-2007].
18. de Kleer, J.: A hybrid truth maintenance system. PARC Technical Report (January 1992)
19. Nayak, P.P.: Automated Modeling of Physical Systems. Springer Verlag LNCS 1003, Cambridge, MA (1995)