Computing Energy-Optimal Tests using DNNF Graphs

Anika Schumann [1], **and Martin Sachenbacher** [2]

[1] *Cork Constraint Computation Centre, University College Cork, Cork, Ireland*
a.schumann@4c.ucc.ie
[2] *Institut für Informatik, Technische Universität München, 85748 Garching, Germany*
sachenba@in.tum.de

ABSTRACT

The goal of testing is to discriminate between multiple hypotheses about a system – for example, different fault diagnoses of an HVAC system – by applying input patterns and verifying or falsifying the hypotheses from the observed outputs. Definitely discriminating tests (DDTs) are those input patterns that are guaranteed to discriminate between different hypotheses of non-deterministic systems. Finding DDTs is important in practice, but can be very expensive ($\sum_2^p$-complete). Even more challenging is the problem of finding a DDT that minimizes the energy consumption of the testing process, i.e., an input pattern that can be enforced with minimal energy consumption and that is a DDT. This paper addresses both problems. We show how we can transform a given problem into a Boolean structure in decomposable negation normal form (DNNF), and extract from it a Boolean formula whose models correspond to DDTs. This allows us to harness recent advances in both knowledge compilation and satisfiability for efficient and scalable DDT computation in practice. Furthermore, we show how we can generate a DNNF structure compactly encoding all DDTs of the problem and use it to obtain an energy-optimal DDT in time linear in the size of the structure.

1 INTRODUCTION

There is a great need to reduce energy consumption whenever possible. This holds in particular for the operation of buildings through electricity, heating, ventilation, and hot water. In industrialized countries the latter constitutes 32% of the total energy use. This percentage is even higher if some of the HVAC components are faulty. It is therefore desirable to identify these faulty components as soon as possible in order to restore the normal operation of the HVAC system. Furthermore it is desirable to minimize the energy consumption of the fault identification process. This implies that energy use should be taken into account during the testing process. Testing involves applying input patterns to a system such that different fault hypotheses about the system can be verified or falsified from the observed outputs.

In many real-world applications of testing, like the testing of HVAC components, the underlying models are non-deterministic, which means that applying an input can lead to several possible outputs. One major source of non-determinism is model abstraction that aggregates the domains of (continuous) system variables into sets of values such as "low", "med", and "high". In such cases the resulting models can no longer be assumed to be deterministic functions, even if the underlying system behavior is deterministic. Another source of non-determinism is the testing situation itself: Even in a rigid environment such as an automotive test-bed, there are inevitably variables or parameters that cannot be completely controlled while the device is being tested.

In the area of diagnosis, Struss (1994) introduced *definitely discriminating tests* (DDTs) for non-deterministic systems modeled as constraints. For a DDT, the sets of possible outputs are disjoint and thus it will necessarily distinguish among hypotheses. Finding DDTs is important in practice because it helps to reduce the number of input patterns that need to be applied and thus allows to quickly identify the faulty component. However, the problem of computing DDTs is in general very expensive ($\sum_2^p$-complete).

Heinz and Sachenbacher (2009) introduced *optimal distinguishing tests* (ODTs), which generalize DDTs by maximizing the ratio of distinguishing over non-distinguishing outcomes. An efficient method for computing ODTs was presented in (Schumann *et al.*, 2009), based on encoding the ODT problem into a Boolean formula, and compiling the formula into a graph in *decomposable negation normal form* (DNNF) (Darwiche, 2001; Darwiche and Marquis, 2002). The DNNF allows efficient derivation of good upper bounds on ratios of model counts, which are then used to prune a systematic search for ODTs.

This method is able to compute DDTs as a special case, and in that role exhibits much better per-

formance than an earlier algorithm by Sachenbacher and Schwoon (2008) specialized for DDTs, as shown in (Schumann *et al.*, 2009). Hence it represents the state of the art in DDT computation. As also shown in (Schumann *et al.*, 2009), however, it can still run out of memory on instances of moderate size, which can be explained by the fact that the final DNNF graph is computed based on a number of auxiliary DNNF graphs, and the compilation can require a multitude of auxiliary variables to be introduced, which leads to slower computation and higher memory requirements.

In this paper, we present a DNNF-based method that is adapted specifically to the problem of computing DDTs, and requires computation of just a single DNNF graph defined over the system variables. This representation is used to transform a quantified Boolean formula defining the DDT problem into a SAT problem that can be solved using an existing SAT solver. This leads to a significantly more efficient method for computing DDTs compared to that of (Schumann *et al.*, 2009).

Furthermore, we present a generalization of DNNF-based test generation that takes into account the energy consumption of enforcing a given input pattern. More specifically, we show how we can compute the values of input variables that can be enforced at minimal energy consumption and that constitute a DDT. Using the algorithm in (Schumann *et al.*, 2010) we achieve this by constructing a DNNF graph, in two steps, that represents all DDTs and allows for the retrieval of an energy-optimal DDT in time linear in the size of the graph. Note, in contrast to the work in (Schumann *et al.*, 2010) we now apply the algorithm to different problems, like HVAC systems, where the aim is to minimize energy consumption during the testing process.

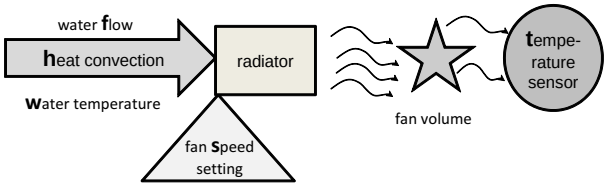
Experimental results from a real-world application show that our method can compute DDTs for instances that were previously intractable, and energy-optimal DDTs where previous approaches could not even compute an arbitrary DDT.

The remainder of the paper is organized as follows. We start with an example of the testing problem, before formally defining DDTs. We then review the notion of DNNF and a generic procedure for energy minimization with DNNF, in the context of our testing problem. Sections 6-8 present our new methods for computing both DDTs and energy-optimal DDTs and Section 9 describes their experimental evaluation. Finally, we conclude with some words on related and future work.

2 EXAMPLE

We start with a small example that motivates and illustrates the problems we are tackling and to which we will refer in the paper.

Consider the simplified model of an HVAC system shown in Figure 1. It consists of two components: a pipe and a radiator. The latter is connected via the pipe to the hot water supply of the building and distributes the thermal energy to the ambient air. In this example we also assume that the radiator contains a fan. The fan power determines how quickly its heat supply



pipe

f	w	h
L	L	L
L	H	H
H	L	L
H	L	H
H	H	H

radiator

h	s	t
L	L	L
L	H	L
H	L	L
H	L	H
H	H	H

broken-pipe

f	w	h
L	L	L
L	H	L
H	L	L
H	L	L
H	H	L

Figure 1: Model of an HVAC system with a possibly faulty pipe component.

leads to a temperature increase and hence to a change in sensor values. In total, there are three controllable input variables (water flow f, water temperature w, fan speed setting s), one internal variable (heat convection h), and one observable output variable (temperature t). As a simplification, we abstract from the actual values for these variables and distinguish only the values high (H) and low (L) for each variable. The models of the two components are given at the bottom of Figure 1:

- the higher the water flow and water temperature, the more heat convection will be transmitted by the **pipe**;
- the higher the heat convection in the pipe, and the higher the fan speed setting of the **radiator** the higher the temperature increase in the room;

However, in case of a **broken-pipe**, no water, i.e. no heat convection is transmitted to the radiator and thus no temperature increase can be observed.

The energy-optimal testing problem for this example is to find a test that can be enforced at minimal energy consumption and that will determine with certainty whether the pipe is broken, i.e., which of the following two hypotheses holds: $M_1 = \{\text{pipe, radiator}\}$, $M_2 = \{\text{broken-pipe, radiator}\}$. For this example, it can be assumed that enforcing a value L requires always less energy than enforcing a value H. Hence the test requiring the lowest energy consumption is $f = \text{L}$, $w = \text{L}$, and $s = \text{L}$. However, this is not a DDT because when stimulating the system with these input values we will certainly observe no temperature increase for both of the hypotheses and thus will not be able to distinguish them. Indeed, there are only two DDTs in this case: (L,H,H) and (H,H,H) (where we will observe a temperature increase iff the pipe is not broken). Hence, (L,H,H) is the energy-optimal DDT for this example.

3 ENERGY-OPTIMAL DEFINITELY DISCRIMINATING TESTS (DDTS)

Following the framework in (Heinz and Sachenbacher, 2009; Schumann *et al.*, 2009), we assume that the sys-

tem can be modeled as a *constraint satisfaction problem* (CSP), which is a triple $M = (\mathcal{V}, \mathcal{D}, \mathcal{C})$, where $\mathcal{D} = D(v_1) \times \ldots \times D(v_n)$ are the finite domains of finitely many variables $v_j \in \mathcal{V}$, $j = 1, \ldots, n$, and $\mathcal{C} = \{C_1, \ldots, C_m\}$ is a finite set of constraints with $C_i \subseteq \mathcal{D}$, $i = 1, \ldots, m$. We denote by X the set of all solutions, that is, assignments $\vec{x} \in \mathcal{D}$ to the variables such that all constraints are satisfied. More formally, $X = \{\vec{x} \mid \vec{x} \in \mathcal{D},\ \mathcal{C}(\vec{x})\}$, where $\mathcal{C}(\vec{x})$ denotes $\forall i\ \vec{x} \in C_i$.

In addition, the system under investigation defines a set of *controllable* (input) variables $\mathcal{I}$ and a set of *observable* (output) variables $\mathcal{O}$. Formally, a hypothesis M for a system is then a CSP whose variables are partitioned into $\mathcal{V} = \mathcal{I} \cup \mathcal{O} \cup \mathcal{L}$, such that $\mathcal{I}$ and $\mathcal{O}$ are the input and output variables of the system, and for all assignments to $\mathcal{I}$, the CSP is satisfiable. The remaining variables $\mathcal{L} = \mathcal{V} \setminus (\mathcal{I} \cup \mathcal{O})$ are called internal state variables. We denote by $\mathcal{D}(\mathcal{I})$ and $\mathcal{D}(\mathcal{O})$ the cross product of the domains of the input and output variables, respectively: $\mathcal{D}(\mathcal{I}) = \times_{v \in \mathcal{I}} D(v)$ and $\mathcal{D}(\mathcal{O}) = \times_{v \in \mathcal{O}} D(v)$.

The goal of testing is then to find assignments of the input variables $\mathcal{I}$ that will cause different assignments of output variables $\mathcal{O}$ for different hypotheses. In the general case of non-deterministic systems, an input assignment can yield more than one possible output assignments. In order to capture sets of outputs, for a given hypothesis M and an assignment $\vec{t} \in \mathcal{D}(\mathcal{I})$ to the input variables, we define the *output function* $\mathcal{X} : \mathcal{D}(\mathcal{I}) \rightarrow 2^{\mathcal{D}(\mathcal{O})}$ with $\vec{t} \mapsto \{\vec{y} \mid \vec{y} \in \mathcal{D}(\mathcal{O}), \exists\, \vec{x} \in X : \vec{x}[\mathcal{I}] = \vec{t} \wedge \vec{x}[\mathcal{O}] = \vec{y}\}$, where $2^{\mathcal{D}(\mathcal{O})}$ denotes the power set of $\mathcal{D}(\mathcal{O})$, and $\vec{x}[\mathcal{I}]$ and $\vec{x}[\mathcal{O}]$ denote the restriction of the vector $\vec{x}$ to the input variables $\mathcal{I}$ and output variables $\mathcal{O}$, respectively. Note that since M will always yield an output, $\mathcal{X}(\vec{t})$ is always non-empty.

A DDT is an assignment to the input variables such that the sets of observable possible outputs for the different hypotheses are disjoint:

Definition 1 (Definitely Discriminating Test) *Consider $n \in \mathbb{N}$ hypotheses $M_1, \ldots, M_n$ with input variables $\mathcal{I}$ and output variables $\mathcal{O}$. Let $\mathcal{X}_i$ be the output function of hypothesis M_i. An assignment $\vec{t} \in \mathcal{D}(\mathcal{I})$ is a* definitely discriminating test *(DDT) if*
$\forall i\, \forall j \neq i\ \mathcal{X}_i(\vec{t}) \cap \mathcal{X}_j(\vec{t}) = \emptyset$.

For testing with non-deterministic automaton models instead of logical theories or CSPs, there exists the analogous notion of *strong distinguishing sequences* (Alur *et al.*, 1995; Boroday *et al.*, 2007). These are input sequences for a non-deterministic finite state machine, such that based on the generated outputs, one can determine the internal state either for some or all feasible runs of the machine. Finding such sequences with a length bounded by some $k \in \mathbb{N}$ can be reduced to the problem of finding DDTs, by unrolling the automaton into a constraint network using k copies of the automaton's transition and observation relation (Esser and Struss, 2007). Hence in this paper we restrict ourselves to models given as (static) networks of finite-domain constraints.

In this paper we do not only consider the problem of finding arbitrary DDTs but also the problem of finding DDTs that can be enforced with minimal energy consumption:

Definition 2 (Energy-Optimal Definitely Discriminating Test) *Let T be the set of all DDTs discriminating the hypotheses $M_1, \ldots, M_k$ and let $e(c, v)$ denote the energy consumption for enforcing value $c \in \mathcal{D}(v)$ on variable v. Further let $E(\vec{t}) = \sum_{h=1}^{l} e(c_h, v_h)$ denote the energy consumption of a test vector $\vec{t} = (v_1 = c_1, v_2 = c_2, \ldots, v_l = c_l)$. A test vector $\vec{t} \in T$ is called* energy-optimal DDT *iff*

$$E(\vec{t}) = min_i \{E(\vec{t_i}) \mid \vec{t_i} \in T\}.$$

4 DECOMPOSABLE NEGATION NORMAL FORM (DNNF)

We briefly review graph-based representations of propositional theories (Darwiche and Marquis, 2002). A propositional theory is in negation normal form (NNF) if it only uses conjunction (and, $\wedge$), disjunction (or, $\vee$), and negation (not, $\neg$), and negation only appears next to variables. An NNF is *decomposable* if the conjuncts of every AND-node share no variables. A DNNF (decomposable NNF) is *smooth* if the disjuncts of every OR-node mention the same set of variables. Such a graph G represents each of its models by a subgraph G_s that satisfies the properties:

- every OR-node in G_s has exactly one child,
- every AND-node in G_s has the same children as it has in G, and
- G_s has the same root as G.

Henceforth the term *subgraph* always denotes a graph satisfying all of the above properties.

DNNF graphs can be generated for propositional theories in conjunctive normal form (CNF) using the publicly available C2D compiler (Darwiche, 2005), and smoothness can be ensured with negligible overhead. The complexity of this compilation is polynomial in the number of variables and exponential only in the treewidth of the system in the worst case. Figure 2 illustrates a smooth DNNF graph representing the set of DDTs for the example shown in Figure 1.[1] This DNNF is composed of two subgraphs consisting of the nodes `A1, O2, A3, -f, w, s` and `A1, O2, A4, f, w, s` respectively.

5 ENERGY MINIMIZATION WITH DNNF

Given a smooth DNNF representation of a propositional theory it is possible to obtain an energy-optimal model in time linear in the size of the DNNF as shown in Algorithm 1.[2] It takes as inputs a smooth DNNF graph and the energy consumption values $e(l)$ for each of its literals. The energy-optimal model, i.e., the subgraph for which $\sum_i\ e(l_i)$ is the lowest, is obtained by

[1] Note that this graph is also *deterministic*, i.e., the disjuncts of every OR-node are pairwise logically inconsistent. Although not required in this work, this property is always enforced by C2D.

[2] This is a variation of the minimization procedure described in (Darwiche, 2001), which labels leaves with 0 and 1 only.

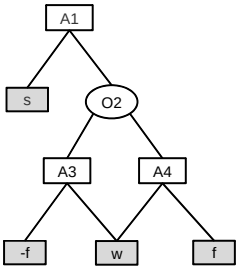


Figure 2: DNNF representing set of DDTs for the example in Figure 1. “A" and “O" indicate AND- and OR-node, respectively. Numbers in non-leaf nodes are their identifiers.

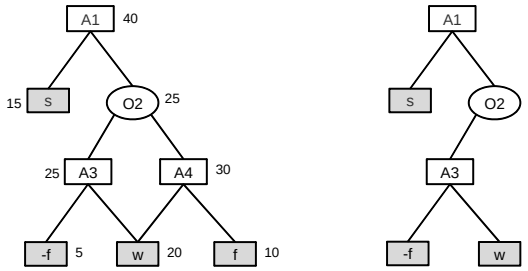


Figure 3: Propagation of energy consumption values for the DNNF graph shown in Figure 2 on the left and an energy-optimal subgraph on the right.

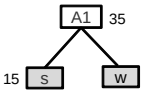
a bottom-up propagation of energy values where each OR-node is labeled by the lowest value of its children and each AND-node is labeled by the sum of the values of its children.

For example, let us assume the following energy consumptions for enforcing values of the input variables in the example of Figure 1:

```
e(-f) = 5    e(-w) = 10    e(-s)=7
 e(f) = 10    e(w) = 20     e(s)=15
```

where $e(-x)$ (resp. $e(x)$) denotes the energy consumption for enforcing the value L (resp. H) on variable x. An energy-optimal model for this example can then be found by propagating these energy consumption values using Algorithm 1 as shown in Figure 3 (left). This model is given by the leaves of an *energy-optimal subgraph*. The latter graph is retrieved by traversing the DNNF top-down such that for each OR-node a child with the lowest value is kept and for each AND-node all children are kept. Figure 3 (right) illustrates the energy-optimal subgraph. Hence an energy-optimal model for this example is $(-f, w, s)$. Its energy consumption is retrieved from the label of the root, `40` in this case.

Note that an energy-optimal subgraph may only give a partial instantiation of the inputs, for example:



In that case an energy-optimal DDT is obtained by simply enforcing the lowest value on each remaining variable, and its energy consumption is obtained by adding the energy consumptions of these enforced values to the label of the root of the subgraph.

Algorithm 1 Energy Minimization with Smooth DNNF

$$e(N) = \begin{cases} e(l) & \text{if } N \text{ is a leaf node representing literal } l \\ \min_i \ e(N_i) & \text{if } N = \bigvee_i N_i \\ \sum_i e(N_i) & \text{if } N = \bigwedge_i N_i \end{cases}$$

6 BOOLEAN FORMULATION OF DDTS

In order to exploit the above efficient minimization procedure for retrieving an energy-optimal DDT, we aim to compile the DDT problem into smooth DNNF. Intuitively, this requires a proposition theory that encodes input vectors such that it is not possible for the corresponding outputs based on the different hypotheses to overlap. Such a theory in DNNF can be obtained in a number of steps, described next.

Since each hypothesis M_i is a CSP, we may assume that each M_i is given as a logical theory in conjunctive normal form (CNF) via any known translation of CSP to SAT (Walsh, 2000). Let M_1 and M_2 be the two hypotheses we wish to distinguish about a system with $(\mathcal{I}, \mathcal{O}, \mathcal{L})$ as input, output, and internal variables. As pointed out in (Sachenbacher and Schwoon, 2008), the existence of a DDT is characterized by the following quantified Boolean formula:

$$\exists \mathcal{I} \ \forall \mathcal{O} \ \forall \mathcal{L} \ \neg (M_1 \wedge M_2),$$

or equivalently,

$$\exists \mathcal{I} \ \neg (\exists \mathcal{O} \ \exists \mathcal{L} \ (M_1 \wedge M_2)),$$

which states that there is an input vector such that the system cannot behave in a way consistent with both hypotheses.

Removing the quantifiers and enlisting the projection operation, we obtain the following propositional theory (denoted F) over the input variables $\mathcal{I}$ only:

$$F = \neg(\Pi_{\mathcal{I}}(M_1 \wedge M_2)),$$

where $\Pi_{\mathcal{I}}(X)$ is the projection of theory X on variables $\mathcal{I}$. The models of theory F are precisely the set of all DDTs.

7 COMPUTING DDTS VIA SAT

We are now ready to tackle the two tasks described earlier: computing DDTs, and computing energy-optimal DDTs. The first requires one call to a DNNF compiler followed by one call to a SAT solver.

Since projection can be performed in linear time on DNNF (Darwiche, 2001), we already make use of DNNF compilation for computing $\Pi_{\mathcal{I}}(M_1 \wedge M_2)$ (i.e., $\neg F$). A single call to a DNNF compiler then suffices to

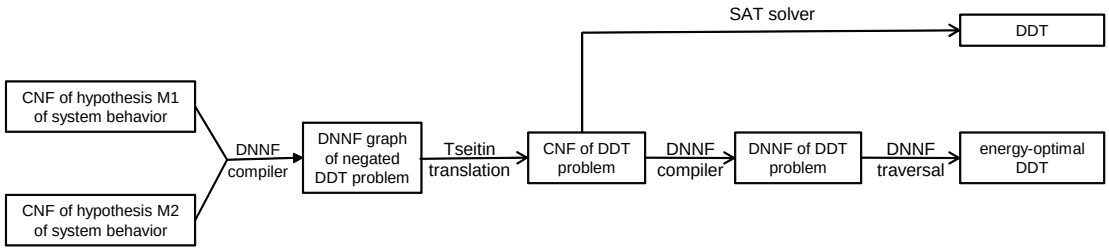


Figure 4: Overview of methods for DDTs and energy-optimal DDTs.

put $\neg F$ in DNNF. This leverages the power of DNNF compilation to obtain a compact representation of a theory over the input variables $\mathcal{I}$ only.

Figure 5 (left) depicts a result of this compilation, a DNNF graph $\mathcal{G}_{\bar{F}}$,[3] for the example in Figure 1. Note that this graph does not need to be smooth, while the one described in the next section does.

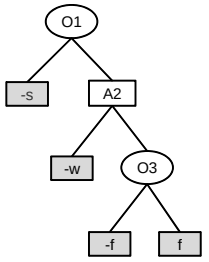
What we actually need to pass on to a SAT solver is the negation of this graph, in CNF. This can be obtained using the well-known Tseitin translation (Tseitin, 1968) coupled with DeMorgan's laws, where the resulting CNF will contain auxiliary variables but have a size polynomial in the DNNF size. Tseitin's translation consists of three steps:

- introduction of one auxiliary variable for each non-leaf node of the DNNF,
- generation of clauses associated with each new variable,
- collection of all clauses into a single CNF with an additional constraint that forces the root to be true.

With our example graph $\mathcal{G}_{\bar{F}}$, we will introduce three new variables: $O1, A2, O3$. Now if we impose negation at the root, apply DeMorgan's laws to push negations down to leaves, and finally apply Tseitin's translation, we obtain:

$$\begin{aligned} O3 &\leftrightarrow (f \wedge \neg f) \\ A2 &\leftrightarrow (w \vee \neg O3) \\ O1 &\leftrightarrow (s \wedge \neg A2) \\ O1 &\leftrightarrow true \end{aligned}$$

The corresponding clauses are shown in Figure 5 (right) and serve as a CNF encoding of the DDT problem, ready to be passed to a SAT solver.



$$\{\neg O3\}$$
$$\{\neg A2, w, \neg O3\}$$
$$\{A2, \neg w\} \quad \{A2, O3\}$$
$$\{O1, \neg s, A2\}$$
$$\{\neg O1, s\} \quad \{\neg O1, \neg A2\}$$
$$\{O1\}$$

Figure 5: DNNF representing the non-DDTs from which the clauses on the right are derived, which encode the DDTs.

8 COMPUTING ENERGY-OPTIMAL DDTS

Models of the CNF encoding described in the previous section correspond to the set of DDTs, and given sufficient computational resources a SAT solver will be able to find one if it exists.

To enable efficient energy minimization, we now invoke once more the DNNF compiler to turn this CNF into smooth DNNF. This graph then represents exactly the set of all DDTs. Thus Algorithm 1 can be applied to efficiently obtain an energy-optimal DDT (or the set of all energy-optimal DDTs if desired). See Figure 3 for an example run of this algorithm.

Note, that by relabeling the nodes of $\mathcal{G}_{\bar{F}}$ according to DeMorgan's laws as described in (Darwiche and Marquis, 2002) we can straight forwardly obtain a graphical representation of the DDT problem in linear time. However, since this graph is not guaranteed to be decomposable it cannot be used to find an energy-optimal test in linear time.

Figure 4 summarizes and illustrates the overall high-level procedures for the two new DDT computation methods. In a nutshell, we are able to solve the energy-optimal DDT problem with two DNNF compilations and two linear-time procedures (one to generate a CNF for the negation of $\mathcal{G}_{\bar{F}}$ and one to retrieve an energy-optimal DDT). Recall that the complexity of the DNNF compilation is polynomial in the number of variables and exponential only in the treewidth of the system in the worst case. Hence the applicability of our approach in practice does not so much depend on the size of the problem but on the structure of the system under test.

9 EXPERIMENTAL EVALUATION

We evaluated our new testing methods on a model of an automotive engine test-bed (Luo *et al.*, 2007), which consists of a throttle, a pipe, and an engine component.

The model was originally given in the form of a mixed discrete-continuous model. It has been turned into a set of 40 discrete problem instances of varying sizes (i.e. variable domains) by applying domain abstractions of different granularity to the continuous variables in the model, using the method described in (Lunze and Nixdorf, 2001), and a direct encoding from

[3] We have kept the trivial OR-node ($\neg f \vee f$) in the graph for illustration purposes.

CSP to SAT (Walsh, 2000).

The goal is then, for each of the instances, to find leaks in the pipe component by assigning three to four controllable variables, and observing three to four measurable variables.

inst.	ODT	eoDDT	DDT
1	0.06	0.03	0.01
2	0.07	0.03	0.01
3	0.10	0.04	0.02
4	17.7	0.42	0.05
5	396	1.7	0.17
6	1566	7.4	0.44
7	-	19.8	0.82
8	-	35.1	1.31
9	-	154	2.42
10	-	213	3.78
11	-	250	5.94
12	-	502	8.18
13	-	880	11.7
14	-	1292	16.8
15	-	1969	24.6
16	-	-	35.4
17	-	-	33.0
18	-	-	42.1
19	-	-	58.0
20	-	-	65.8

inst.	DDT
21	84.5
22	110
23	146
24	167
25	191
26	246
27	286
28	345
29	432
30	518
31	615
32	670
33	792
34	914
35	1056
36	1139
37	1329
38	1492
39	1724
40	1894

Table 1: Computation times in sec applying the ODT, energy-optimal DDT (eoDDT), and DDT approach to different instances of the example application.

The experiments were performed on a Linux Dual-Core PC with 1GB of RAM using the DNNF compiler C2D (Darwiche, 2005) and the SAT solver Tinisat (Huang, 2007). Table 1 shows the computation times for finding a DDT using the previous ODT method (Schumann *et al.*, 2009), an energy-optimal DDT using our new method (two compilations), and a DDT using our new method (compilation+SAT). Blank entries signify failure due to memory exhaustion.

The results indicate improvements of several orders of magnitude over the previous state of the art. For example, the largest instance solved by ODT (# 6) took it 1566 seconds, but took our new DDT method only 0.44 seconds, and we were able to find an energy-optimal DDT in 7.4 seconds. The improvement allowed much larger instances to be solved with both our new methods, and particularly with the compilation+SAT method for DDTs.

As we briefly discussed in the introduction, the poorer performance of the ODT method appears to be partly due to the more complex DNNF compilations required. Figure 6 illustrates this issue in more concrete terms, where the number of nodes in the largest DNNF graph generated is shown for each instance, for the ODT method and our new DDT method.

For example, for instance 6 the DNNF has already over 100,000 nodes for the ODT method, but less than 3,000 nodes for the new method. Figure 6 does not

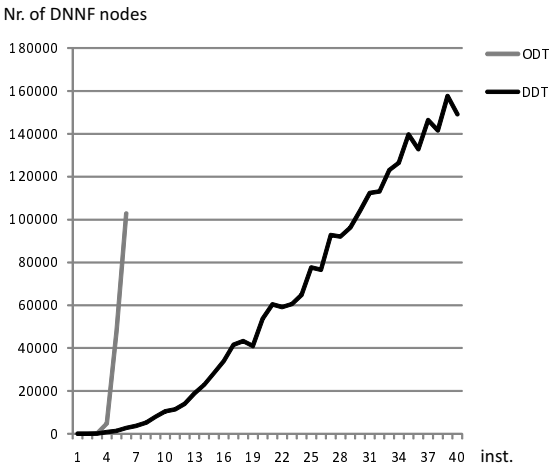


Figure 6: Number of nodes of the largest DNNF generated for each of the instances in Table 1.

show DNNF sizes for the second compilation required by our method for energy-optimal DDTs, but they are always smaller than in the first compilation except in the case involving instance 1.

Finally, that our method for DDTs scales much better than for energy-optimal DDTs is consistent with the difference in complexity between SAT and compilation. For an illustration of this difference, consider the fact that compilations produced by the C2D compiler are known to permit model counting in linear time (Darwiche and Marquis, 2002)—model counting is known to be #P-complete while SAT is NP-complete.

10 RELATION TO PREVIOUS WORK

The problem of computing definitely discriminating tests in the area of diagnosis was first introduced in (Struss, 1994). Struss (2007) then shows how to optimize the testing process by reducing the number of tests that need to be carried out to discriminate a set of fault hypotheses. In contrast to that work our approach seeks to directly compute optimized individual tests that discriminate a set of hypotheses. This motivation is similar to that of the works on computing optimal distinguishing tests (Heinz and Sachenbacher, 2009; Schumann *et al.*, 2009). However, while the latter methods aim at computing tests that will discriminate hypotheses most likely our approach targets problems for which there exist tests that definitely discriminate hypotheses. As the experimental results have shown this specialized problem can be solved much faster, even when we optimize the tests with respect to energy consumption.

In automata theory, Alur *et al.* (1995) studied the analogous problem of generating *strong distinguishing sequences* for non-deterministic finite state machines, which for sequences of bounded length can be reduced to that of finding DDTs (Esser and Struss, 2007). The work of Torta and D. Theseider Dupré (2008) then also considers the cost for testing such systems. Here the

authors aim at discriminating a set of hypotheses and assume that the available actions will necessarily distinguish between them. In contrast to that work we have considered the testing of static nondeterministic systems where different actions will not necessarily discriminate between hypotheses.

From a conceptual point of view the work in (Darwiche, 1998) is the closest to ours. It exploits the benefits of DNNF compilation for the fault diagnosis of systems. First it computes the DNNF graph representing the set of all diagnosis and then it uses a minimization procedure to extract all minimal ones. Thus the aim of this method is to identify all minimal diagnoses that are consistent with a set of observations. However, in practice, the resulting set of diagnoses, i.e. of fault hypotheses, can be very large. Our approach complements this work. It aims at tackling this problem by providing a procedure for discriminating between different hypotheses, like these fault hypotheses. It thus needs to consider not only the observable and faulty variables of a system but also the controllable ones. We have shown how to exploit the distinct properties of the DDT problem by solving it based on DNNF graphs that are defined over input variables only. Using a variation of the minimization procedure in (Darwiche, 1998) we were also able to identify the actions, i.e. the assignments to controllable input variables that can perform the discrimination of fault hypotheses and that can be carried out with minimal energy consumption.

11 CONCLUSION AND FUTURE WORK

We have presented a new algorithm for computing DDTs that exhibits much better efficiency and scalability than previous approaches on a realistic benchmark. This was achieved by constructing a Boolean formula that encodes the complex DDT problem ($\sum_2^p$-complete) and that can be transformed into a SAT problem via DNNF compilation, which is often very efficient for real-world problems that have structure.

In addition we have studied the problem of computing DDTs that can be enforced at minimal energy consumption, a problem which is very relevant in practice but has not been considered before. We proposed an algorithm for this problem that involves one additional step of DNNF compilation such that energy-optimal DDTs can be retrieved from the final DNNF in time linear in the size of the DNNF. With this algorithm we were able to compute energy-optimal DDTs for instances where previous methods could not even find an arbitrary DDT.

The success of our new methods can be partly ascribed to the fact that they are better able to exploit structural properties of the system (compared with Sachenbacher and Schwoon 2008), or that they require less complex DNNF compilations (compared with Schumann *et al.* 2009).

Topics for future work include extending our approach to systems that admit possibly discriminating tests but not DDTs, and to systems whose nondeterminism is quantified by a probabilistic model. It would also be interesting to analyze to what extent incremental DNNF compilation techniques like (Venturini and Provan, 2008) could be used for the test generation of even larger systems.

ACKNOWLEDGMENTS

This work was supported by the Science Foundation Ireland under the ITOBO Grant No. 05/IN/I886, and the Deutsche Forschungsgemeinschaft under grant SA 1000/2-1.

REFERENCES

(Alur *et al.*, 1995) R. Alur, C. Courcoubetis, and M. Yannakakis. Distinguishing tests for nondeterministic and probabilistic machines. In *Proceedings of the ACM Symposium on Theory of Computing*, pages 363–372, 1995.

(Boroday *et al.*, 2007) S. Boroday, A. Petrenko, and R. Groz. Can a model checker generate tests for non-deterministic systems? *Electronic Notes in Theoretical Computer Science*, 190:3–19, 2007.

(Darwiche and Marquis, 2002) A. Darwiche and P. Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

(Darwiche, 1998) Adnan Darwiche. Model-based diagnosis using structured system descriptions. *Journal of Artificial Intelligence Research*, 8:165–222, 1998.

(Darwiche, 2001) A. Darwiche. Decomposable negation normal form. *Journal of the ACM*, 48(4):608–647, 2001.

(Darwiche, 2005) A. Darwiche. The C2D compiler user manual. Technical report, Comp. Sci. UCLA, 2005.

(Esser and Struss, 2007) M. Esser and P. Struss. Fault-model-based test generation for embedded software. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 342–347, 2007.

(Heinz and Sachenbacher, 2009) S. Heinz and M. Sachenbacher. Using model counting to find optimal distinguishing tests. In *Proceedings of the Sixth International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CPAIOR)*, pages 117–131, 2009.

(Huang, 2007) J. Huang. A case for simple SAT solvers. In *Proceedings of the 13th International Conference on Principles and Practice of Constraint Programming (CP)*, pages 839–846, 2007.

(Lunze and Nixdorf, 2001) J. Lunze and B. Nixdorf. Representation of hybrid systems by means of stochastic automata. *Mathematical and Computer Modeling of Dynamical Systems* , 4:383–422, 2001.

(Luo *et al.*, 2007) J. Luo, K. Pattipati, L. Qiao, and S. Chigusa. An integrated diagnostic development process for automotive engine control systems. *IEEE Trans. on Systems, Man, and Cybernetics*, 37(6):1163–1173, 2007.

(Sachenbacher and Schwoon, 2008) M. Sachenbacher and S. Schwoon. Model-based testing using quantified CSPs. In *ECAI-08 Workshop on Model-based Systems*, 2008.

(Schumann *et al.*, 2009) A. Schumann, M. Sachenbacher, and J. Huang. Constraint-based optimal testing using DNNF graphs. In *Proceedings of the 15th International Conference on Principles and Practice of Constraint Programming (CP)*, pages 731–745, 2009.

(Schumann *et al.*, 2010) A. Schumann, J. Huang, and M. Sachenbacher. Computing cost-optimal definitely discriminating tests. In *Proceedings of the 24th National Conference on Artificial Intelligence (AAAI)*, 2010.

(Struss, 1994) P. Struss. Testing physical systems. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, pages 251–256, 1994.

(Struss, 2007) P. Struss. Model-based optimization of testing through reduction of stimuli. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 593–598, 2007.

(Torta and D. Theseider Dupré, 2008) G. Torta and L. Anselma D. Theseider Dupré. Hypothesis discrimination with abstractions based on observation and action costs. In *Proceedings of the 19th International Workshop on Principles of Diagnosis (DX-08)*, pages 189–196, 2008.

(Tseitin, 1968) G. Tseitin. On the complexity of derivation in propositional calculus. *Studies in Constructive Mathematics and Mathematical Logic*, pages 115–125, 1968.

(Venturini and Provan, 2008) A. Venturini and G. Provan. Incremental algorithms for approximate compilation. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI-08)*, 2008.

(Walsh, 2000) T. Walsh. SAT vs. CSP. In *Proceedings of the Sixth International Conference on Principles and Practice of Constraint Programming (CP)*, pages 441–456, 2000.