# Finding Explanations: an Empirical Evaluation of Abductive Diagnosis Algorithms

**Roxane Koitz*** and **Franz Wotawa**
Institute for Software Technology
Graz University of Technology, Graz, Austria
e-mail: {rkoitz, wotawa}@ist.tugraz.at

## Abstract

Abductive inference provides consistent explanations for observable effects and has been of special interest in the context of diagnosis. The abduction problem is in general NP-hard, thus, there is a high motivation to derive solutions efficiently for practical instances. In this paper, we focus on propositional abduction in the framework of model-based diagnosis. We review four algorithms to compute explanations: one employs an ATMS to derive diagnoses and the others are conflict-directed methods based on an unsatisfiable reformulation of the abductive system description. In an empirical evaluation we compare the different approaches on practical examples. Our experiments indicate that the ATMS provides the best performance results for the majority of problems.

## 1 Introduction

Abductive inference, as a form of non-monotonic reasoning, attempts to derive a set of causes which best explain an effect. Within this paper we focus on logic-based abduction, which is formulated as finding a consistent set of hypotheses implying a given observation together with the background knowledge. A variety of approaches, such as consequence finding [Marquis, 2000] or proof-tree completion [McIlraith, 1998], have been proposed as methods for mechanizing abductive reasoning and lead to the development of several systems, e.g. DART [Genesereth, 1984] or Theorist [Poole *et al.*, 1987]. In the context of logic programming, abductive logic programming [Kakas *et al.*, 1992] emerged aiming at providing a framework and set of techniques for performing abductive reasoning [Denecker and De Schreye, 1998; Van Nuffelen, 2001]. It is well known that abduction is in general an NP-hard problem with potentially an exponential number of solutions [Bylander *et al.*, 1991]. Thus, there is a demand to compute abductive explanations efficiently for instances of practical size and complexity.

*Authors are listed in alphabetical order.

Even though abduction has been performed in the context of various tasks [Denecker and Kakas, 2002], such as planning [Poole and Kanazawa, 1994] or natural language processing [Ovchinnikova *et al.*, 2014], its prevalent application is in diagnosis. Model-based diagnosis has been proposed as an improvement to fault localization and relies on a formal system description encompassing structural as well as behavioral knowledge of the physical artifact. Within the last decades an extensive body of research has distinguished two logical definitions: consistency-based and abductive diagnosis. Consistency-based diagnosis relies on a formalization of the correct system response and identifies failures through inconsistency [Reiter, 1987]. In contrast, the abductive approach employs models of faulty behavior to reason from symptoms to causes and is based on a stricter criteria as it finds consistent explanations entailing the observations [Console *et al.*, 1991]. Abductive model-based diagnosis has been applied, e.g., to environmental decision support systems [Wotawa *et al.*, 2009].

The computation of explanations has not only been studied in the context of diagnosis, but also has received attention in the field of constraint satisfaction problems and infeasibility analysis. Junker [2004] describes an algorithm generating preferred explanations for over-constrained systems. By employing a divide and conquer strategy, conflicting constraints can be efficiently computed. These contradictions essentially constitute the causes for the unsatisfiability of the system. Within the field of infeasibility analysis, conflicts refer to Minimal Unsatisfiable Subsets (MUSes). Recently, Liffiton *et al.* [2015] present a direct MUSes approach which exploits the power-set lattice. Many algorithms for computing unsatisfiable cores, however, do not generate them directly, but rely on their hitting set dual Minimal Correction Subsets (MCSes). Liffiton and Sakallah [2008] propose the CAMUS algorithm utilizing this hitting set duality to produce MUSes by first computing all MCSes.

In this paper, we investigate approaches to computing explanations in the framework of abductive model-based diagnosis. In particular, we examine one direct proof method by exploiting an assumption-based truth maintenance system (ATMS) to infer consistent diagnoses. The other techniques utilize the unsatisfiability

of a rewritten system description to derive explanations. The first algorithm determines conflicts based on a hitting set directed acyclic graph (HS-DAG), while the two remaining rely on MUSes and MCSes computation.

The remainder of this paper is structured as follows. Section 2 introduces the theoretical foundations of abductive model-based diagnosis and further provides definitions in the context of unsatisfiable formulas. Subsequently, we describe the selected algorithms and tools. In Section 4 we present the set-up and results of an empirical evaluation, followed by our conclusions.

## 2 Preliminaries

In this section we define the Propositional Horn Clause Abductions Problem, which functions as the basis of our research. Subsequently, we discuss certain subsets of unsatisfiable formulas and show their connection to abductive diagnosis.

### 2.1 Abductive Model-Based Diagnosis

Abductive inference, in the context of formal logic, can be defined as the search for a set of hypotheses which entail the observations, while being consistent in conjunction with the background theory. Model-based diagnosis builds upon a formalization of the system behavior. Thus, abductive model-based diagnosis requires a description of the system response in the presence of a fault in order to compute causes entailing symptoms.

In general logic-based abduction is an intractable problem, however, there exist certain subsets, such as definite propositional Horn theories, where abduction is polynomial [Eiter and Gottlob, 1995; Nordh and Zanuttini, 2008]. We draw on these findings and focus in our research on propositional logic. Note that the following definitions are similar to the ones by Friedrich *et al.* [1990].

**Definition 1 (Knowledge base (KB))** *A knowledge base (KB) is a tuple (A,Hyp,Th) where A denotes the set of propositional variables, Hyp* $\subseteq$ *A the set of hypotheses, and Th the set of Horn clause sentences over A.*

The set of hypotheses contains the propositions which can either be assumed true or false and refer to possible causes. In order to form an abduction problem, in particular a Propositional Horn Clause Abduction Problem, we consider in addition to the knowledge base a set of observations for which explanations are to be computed.

**Definition 2** ***(Propositional Horn Clause Abduction Problem (PHCAP))*** *Given a knowledge base (A,Hyp,Th) and a set of observations Obs* $\subseteq$ *A then the tuple (A,Hyp,Th,Obs) forms a Propositional Horn Clause Abduction Problem (PHCAP).*

**Definition 3 (Diagnosis; Solution of a PHCAP)** *Given a PHCAP (A,Hyp,Th,Obs). A set* $\Delta \subseteq Hyp$ *is a solution if and only if* $\Delta \cup Th \models Obs$ *and* $\Delta \cup Th \not\models \bot$. *A solution* $\Delta$ *is parsimonious or minimal if and only if no set* $\Delta' \subset \Delta$ *is a solution.*

A solution to the PHCAP constitutes an abductive diagnosis as it comprises the set of hypotheses explaining the observations.

*Example 1:* Consider the simplified example of a converter $KB$ of an industrial wind turbine.

$$Hyp = \left\{ \begin{array}{c} mode(Fan, Corrosion), \\ mode(Fan, TMF), mode(IGBT, HCF) \end{array} \right\}$$

$$A = \left\{ \begin{array}{c} P\_turbine, T\_nacelle, mode(Fan, Corrosion), \\ mode(Fan, TMF), mode(IGBT, HCF) \end{array} \right\}$$

$$Th = \left\{ \begin{array}{c} mode(Fan, Corrosion) \rightarrow P\_turbine, \\ mode(Fan, TMF) \rightarrow P\_turbine, \\ mode(IGBT, HCF) \rightarrow T\_nacelle, \\ mode(IGBT, HCF) \rightarrow P\_turbine \end{array} \right\}$$

Let us assume an increased temperature in the nacelle ($T\_nacelle$) and a lower than expected power output ($P\_turbine$) can be observed, i.e. $Obs = \{T\_nacelle, P\_turbine\}$. Thus, the solution to the PHCAP is $\Delta_1 = \{mode(IGBT, HCF)\}$.

To compute the abductive explanations for an observed effect, one can check all subsets of hypotheses to determine whether they entail the observations or not. This approach, however, is computationally expensive and therefore not applicable in a practical setting.

### 2.2 SAT-Based Abduction

We assume standard definitions for propositional logic throughout this section [Chang and Lee, 1973]. If a propositional formula $\phi$ in CNF is unsatisfiable, there are subsets which are of special interest in the context of abduction. In this section we define said sets first and then examine how they can be used for abductive diagnosis. The subsequent definitions are taken from Liffiton and Sakallah [2008].

A Minimal Unsatisfiable Subset (MUS) contains a subset of clauses which cannot be satisfied simultaneously.

**Definition 4** ***(Minimal Unsatisfiable Subset (MUS))*** *A subset* $U \subseteq \phi$ *is an MUS if* $U$ *is unsatisfiable and* $\forall C_i \in U, U \setminus (\{C_i\})$ *is satisfiable.*

Notice that every proper subset of an MUS is satisfiable. Its hitting set dual, Minimal Correction Subset (MCS), comprises clauses that correct the unsatisfiable formula when removed [Birnbaum and Lozinskii, 2003].

**Definition 5** ***(Minimal Correction Subset (MCS))*** *A subset* $M \subseteq \phi$ *is an MCS if* $\phi \setminus M$ *is satisfiable and* $\forall C_i \in M, \phi \setminus (M \setminus \{C_i\})$ *is unsatisfiable.*

Since removing an MCS results in a feasible formula, any MCS is the complement of some Maximal Satisfiable Subset (MSS), which is a generalization of a solution to the MAX-SAT problem.

**Definition 6** ***(Maximal Satisfiable Subset (MSS))*** *A subset* $S \subseteq \phi$ *is an MSS if* $S$ *is satisfiable and* $\forall C_i \in \phi \setminus S, S \cup \{C_i\})$ *is unsatisfiable.*

*Example 2:* Consider the unsatisfiable formula $\phi$ in CNF.

$$\phi = \overbrace{(\neg a \vee \neg b \vee c)}^{C_1} \wedge \overbrace{(\neg c \vee d)}^{C_2} \wedge \overbrace{(c)}^{C_3} \wedge \overbrace{(\neg d)}^{C_4}$$

The combination of clauses $C_2, C_3$ and $C_4$ results in $\phi$ being unsatisfiable, hence the unsatisfiable cores are

$$\text{MUSes}(\phi) = \{\{C_2, C_3, C_4\}\}.$$

Via hitting set computation we derive the following set of MCSes:

$$\text{MCSes}(\phi) = \{\{C_2\}, \{C_3\}, \{C_4\}\}.$$

Removing any MCS from $\phi$ results in the formula being satisfiable. Subsequently, we can compute the Maximal Satisfiable Subsets (MSSes) by forming the complements:

$$\text{MSSes}(\phi) = \{\{C_1, C_3, C_4\}, \{C_1, C_2, C_4\}, \{C_1, C_2, C_3\}\}.$$

As aforementioned, the MUSes correspond to the explanations of an over-constrained system [Junker, 2004]. In order to generate abductive explanations on basis of unsatisfiable formulae, we recast the first condition of Definition 3 of an abductive explanation from $\Delta \cup Th \models Obs$ to $Th \cup \Delta \cup \{\neg Obs\} \models \bot$ by logical equivalence. $\{\neg Obs\}$ consists of the complement of each observation in $Obs$, i.e. $\forall o \in Obs : \neg o \in \{\neg Obs\}$. Thus, computing the abductive explanations is reformulated as the search for a refutation proof comprising propositions from $Hyp$ [McIlraith, 1998]. In other words, we can restate the problem of computing diagnoses to finding the conflict sets of $Th \wedge Hyp \wedge \{\neg Obs\}$ which by definition are equivalent to the MUSes of said formula.

Naturally, MUSes contains several unsatisfiable subsets irrelevant for the diagnostic task. Since we are solely interested in minimal explanations, we dismiss certain subsets and parts of MUSes. We first eliminate all propositions not corresponding to hypotheses. The resulting solution may contain supersets of diagnoses, which we subsequently remove to derive minimal explanations. We refer to MUSes corresponding to parsimonious abductive diagnoses as $MUSes_{Hyp}$.

# 3 Algorithms for Computing Abductive Explanations

In the following, we depict our four approaches to abductive diagnosis based on a propositional logic model. For each method, we give a brief description of the underlying notion for deriving explanations and subsequently discuss specific tools and algorithms included in the empirical evaluation.

## 3.1 ATMS

De Kleer's [1986a] ATMS has been recognized as a general abduction engine for propositional Horn clause sentences [Levesque, 1989]. An ATMS exploits a graph representation of the theory, where hypotheses, observations, and contradiction are vertices. The edges are determined by the implications of the underlying Horn clauses. By assigning a label to each node, the ATMS keeps track of the hypotheses from which each vertex can be inferred from. Specifically, a label is a set of sets of hypotheses. Whenever a new rule is applied to the ATMS, the nodes' labels are updated, consistency is ensured, and valid explanations for a given effect can be directly determined. Notice that the labels can grow exponentially in the number of assumptions [de Kleer, 1986b].

Wotawa *et al.* [2009] propose Algorithm `abductive-Explanations`, which computes abductive diagnoses for a given PHCAP by exploiting an ATMS. After passing the Horn clauses composing the theory to the ATMS, a single implication is added: $o_1 \wedge o2 \wedge \ldots \wedge o_n \rightarrow obs$, where $\{o_1, o_2, \ldots, o_n\}$ correspond to the observations and $obs$ denotes a new proposition not yet considered in $A$. The label of $obs$ comprises all hypotheses which inferred the observations, thus constitute the solutions to the PHCAP. Since the ATMS terminates due to a finite number of hypotheses, the Algorithm `abductiveExplanations` is guaranteed to halt as well. We utilized a Java implementation of `abductiveExplanations` for our empirical evaluation.

**Algorithm 1** abductiveExplanations [Wotawa *et al.*, 2009]

**procedure** ABDUCTIVEEXPLANATIONS $(A, Hyp, Th, Obs)$
  Add $Th$ to $ATMS$
  Add $\left(\bigwedge_{o \in Obs} o \rightarrow obs\right)$ to $ATMS$ ▷ $obs \notin A$
  **return** the label of $obs$
**end procedure**

## 3.2 Conflict-Driven Search via HS-DAG

By detecting a discrepancy between the predicted and actual behavior, i.e. a conflict, Reiter [1987] derived consistency-based diagnoses via minimal hitting set computation. A conflict arises when, under the assumption all components are behaving correctly, an observation is inconsistent with the expected performance. Thus, conflicts correspond to hypotheses contradicting observations. By rewriting the abductive model, as noted in Section 2.2, we can derive conflicts which constitute abductive diagnoses.

Reiter's approach maintains a tree to compute all minimal hitting sets based on conflicts. These conflicts can be generated on demand by applying a theorem prover, which returns a refutation involving hypotheses if one exists. Starting from an initial conflict set as root node, the tree is iteratively extended in a breadth first manner. At each node $n$, labeled with conflict $C$, an outgoing edge $h(n)$ is generated for each $c \in C$. Each edge label is checked for consistency. In case it is consistent the corresponding node determines a leaf and thus a minimal hitting set, otherwise a new conflict set is derived, such that it is disjoint to the current set of edge labels. Several pruning techniques ensure the minimality of the hitting sets and allow the use of non minimal conflicts. Greiner *et al.* [1989] corrected some inadequacies of Reiter's algorithm and devised an approach performing on a directed acyclic graph (DAG) instead of a tree.

Algorithm `hsdagAB` is based on HS-DAG and a theorem prover to derive conflicts and subsequently minimal

abductive explanations. Given a PHCAP, we generate an implication with a conjunction of observations on the left hand side and the contradiction on the right hand side, i.e. $o_1 \wedge o_2 \wedge \ldots \wedge o_n \rightarrow \bot$. The theory $Th$, the implication, and the theorem prover, represented by $TP$, are supplied to HS-DAG. $CONF$ corresponds to the set of conflicts obtained from the hitting set algorithm. Note that these conflicts are not ensured to be minimal; thus, we remove all supersets afterwards.

**Algorithm 2** hsdagAB

```
procedure HSDAGAB(A, Hyp, Th, Obs)
    Δ − Set, CONF ← ∅
    TP ← Th ∪ (⋀_{o∈Obs} o → ⊥)          ▷ Theorem Prover
    CONF ← HS-DAG(TP)                     ▷ HS-DAG
    for all c ∈ CONF do
        if ∄c′ ∈ CONF : c′ ⊆ c then
            Δ − Set ← c
        end if
    end for
    return Δ − Set
end procedure
```

For our evaluation we utilized the publicly available diagnosis engine JDiagengine[1] which implements a conflict-driven search via HS-DAG [Peischl and Wotawa, 2003] exploiting a Horn clause theorem prover [Minoux, 1988]. JDiagengine as well as `hsdagAB` are Java implementations.

### 3.3 Direct MUS Approach

In Section 2.2 we examined the relation between MUSes and abductive diagnoses. By rewriting the model to an unsatisfiable formula, the abduction problem consists in computing the sets of hypotheses which are responsible for the infeasibility, i.e. $MUSes_{Hyp}$.

Algorithm `musAB` employs a MUS enumeration procedure and thereon computes the minimal abductive diagnoses, denoted $MUSes_{Hyp}$. We create an unsatisfiable CNF encoding of the problem denoted $\phi$. Since $Th$ consists of Horn clauses, we can easily convert it into a CNF representation, which we refer to as $\mathcal{T}$. For each $h \in Hyp$, we create a single clause assuming $h$ to be true. Additionally, we generate a disjunction containing the negated observations, i.e. $\neg o_1 \vee \neg o_2 \vee \ldots \vee \neg o_n$.

*Example 1 (cont.):* Consider again our running example of the converter. Let $\phi$ be the unsatisfiable CNF representation of the abduction problem:

$C_1 : \neg mode(Fan, Corrosion) \vee P\_turbine$

$C_2 : \neg mode(Fan, TMF) \vee P\_turbine$

$C_3 : \neg mode(IGBT, HCF) \vee P\_turbine$

$C_4 : \neg mode(IGBT, HCF) \vee T\_nacelle$

$C_5 : mode(Fan, Corrosion)$

$C_6 : mode(Fan, TMF)$

$C_7 : mode(IGBT, HCF)$

[1] http://www.ist.tugraz.at/modremas/index.html

**Algorithm 3** musAB

```
procedure MUSAB(A, Hyp, Th, Obs)
    MUSes, Δ − Set ← ∅
    φ ← T ∪ Hyp ∪ ⋁_{o∈Obs} ¬o
    MUSes ← MUSes(φ)          ▷ MUS enumeration algorithm
    for all m ∈ MUSes do
        M ← m ∩ Hyp
    end for
    for all u ∈ M do
        if ∄u′ ∈ M : u′ ⊆ u then
            MUSes_Hyp ← u
        end if
    end for
    return Δ − Set ← MUSes_Hyp
end procedure
```

$C_8 : \neg T\_nacelle \vee \neg P\_turbine$

Clauses $C_1$ to $C_4$ refer to $\mathcal{T}$, $C_5$ to $C_7$ to the set $Hyp$ and clause $C_8$ contains the negation of the set of observations. We obtain the following MUSes from $\phi$:

$$MUSes = \left\{ \begin{array}{c} \{C_3, C_4, C_7, C_8\}, \{C_1, C_3, C_5, C_7, C_8\}, \\ \{C_2, C_3, C_6, C_7, C_8\} \end{array} \right\}$$

Since we are only interested in the abducibles, we remove all clauses not associated with hypotheses. Let $M$ be the resulting set:

$$M = \{ \ \{C_7\}, \{C_5, C_7\}, \{C_6, C_7\} \ \}.$$

Eliminating all supersets we obtain $MUSes_{Hyp} = \{\{C_7\}\}$. Hence the abductive diagnosis is $\Delta_1 = \{mode(IGBT, HCF)\}$.

We implemented `musAB` in Java and employed the MUS enumeration tool MARCO[2] [Liffiton *et al.*, 2015]. MARCO computes MUSes and MSSes based on an exploration of the power-set lattice. Given an unsatisfiable clause set, all of its supersets are unsatisfiable as well; thus, an MUS defines a "low point" in an infeasible region. Similarly, an MSS characterizes a "high point" in a satisfiable region. In each iteration MARCO investigates an unexplored part of the lattice and traverses through the power-set until either an MUS or an MSS is found. MARCO is implemented in Python using MUSer2[3] and MiniSat[4].

### 3.4 Indirect Approach

Many MUS enumeration algorithms refrain from computing the unsatisfiable cores directly, but exploit its hitting set dual MCS, since finding satisfiable subsets is an NP-complete problem, whereas UNSAT resides in CoNP [Liffiton and Sakallah, 2008]. Therefore, we examine an indirect approach, which first computes the MCSes and then determines the MUSes [Koitz and Wotawa, 2015b]. In the case of diagnosis we are only interested in the hypotheses, which have been used to derive a conflict. Thus, for further computation we select MCS which only contain clauses referring to explanations. We create the

[2] http://sun.iwu.edu/ mliffito/marco/

[3] http://logos.ucd.ie/wiki/doku.php?id=muser

[4] http://minisat.se/

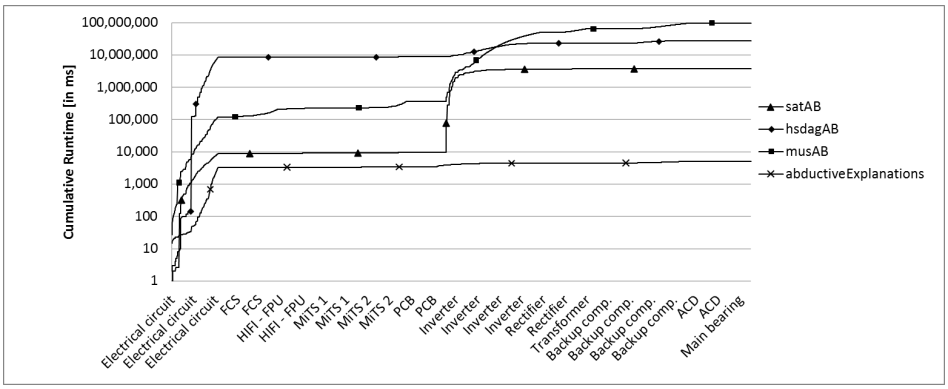


Figure 1: Cumulative runtimes of `abductiveExplanations`, `hsdagAB`, `musAB`, and `satAB` for the experiment.

set $MCSes_{Hyp}$ such that $\forall m \in MCSes_{Hyp} : m \subseteq Hyp$. This has two practical consequences: it reduces the number of sets to be considered by the hitting set algorithm and the corresponding MUSes derived from $MCSes_{Hyp}$ already constitute the abductive diagnoses.

Algorithm `satAB` computes the set of abductive diagnoses for a given PHCAP based on an MCS enumeration algorithm. Note that the unsatisfiable model $\phi$ is the same as in the direct MUS approach.

**Algorithm 4** satAB

**procedure** SATAB $(A, Hyp, Th, Obs)$
  $MCSes, MCSes_{Hyp} \leftarrow \emptyset$
  $\phi \leftarrow \mathcal{T} \cup Hyp \cup \bigvee_{o \in Obs} \neg o$
  $MCSes \leftarrow$ `MCSes(`$\phi$`)` ▷ MCS enumeration algorithm
  **for all** $m \in MCSes$ **do**
    **if** $m \subseteq Hyp$ **and** $m \cup Th$ is consistent **then**
      $MCSes_{Hyp} \leftarrow m \cup MCSes_{Hyp}$
    **end if**
  **end for**
  $\Delta - Set \leftarrow$ `MHS(`$MCSes_{Hyp}$`)` ▷ Minimal hitting set algorithm
  **return** $\Delta - Set$
**end procedure**

*Example 1 (cont.):* Computing the $MCSes$ of $\phi$ we obtain:

$$MCSes = \left\{ \begin{array}{c} \{C_3\}, \{C_7\}, \{C_8\}, \{C_4, C_5, C_6\}, \\ \{C_2, C_4, C_5\}, \{C_1, C_4, C_6\}, \{C_1, C_2, C_4\} \end{array} \right\}.$$

Extracting the MCSes, which only contain clauses from $Hyp$ and are consistent with regard to the theory, results in $MCSes_{Hyp} = \{\{C_7\}\}$. By computing the hitting set of $MCSes_{Hyp}$, we obtain the set of MUSes solely referring to explanations, which are in fact the set of abductive diagnoses. In our example $\Delta_1 = \{mode(IGBT, HCF)\}$.

For our evaluation we implemented `satAB` in Java and utilized the $\text{MCS}_{\text{LS}}$[5] tool by Marques-Silva *et al.* [2013] as the MCS computation procedure . $\text{MCS}_{\text{LS}}$ is written in C++, employs MiniSat[6], and provides the possibility to apply several MCS enumeration algorithms. We decided for the CLD approach of $\text{MCS}_{\text{LS}}$, which takes advantage of disjoint unsatisfiable cores. Regarding the hitting set computation, we engaged a Java implementation of the Binary Hitting Set Tree algorithm [Lin and Jiang, 2003] which performed well in a comparison of minimal hitting set algorithms [Pill *et al.*, 2011].

[5]http://logos.ucd.ie/web/doku.php?id=mcsls
[6]http://minisat.se/

## 4 Empirical Evaluation

In this section, we describe our empirical evaluation setup and report on the obtained results. All the numbers presented in this section were obtained from a Lenovo ThinkPad T540p Intel Core i7-4700MQ processor (2.60 GHz) with 8 GB RAM running Ubunutu 14.04 (64-bit).

We generated propositional Horn models from several Failure Mode Effect Analyses covering various technical systems by utilizing a mapping function. A detailed description of the conversion process can be found in Wotawa [2014] and Koitz and Wotawa [2015a]. Table 2 provides an overview of the models' structure as well as some characteristics of the problem instances. It is worth noting that the system descriptions vary in the number of hypotheses (Hyp), possible observables (Obs), and implications (Th). Due to theory comprising Horn clauses, a conversion into a CNF representation, suitable for the MUS-based and MCS-based computation, is straightforward.

In the experiments, we computed the abductive explanations for $|Obs|$ from one to the maximum number of effects possible. The observations were generated randomly; however, the same set was used for all algorithms. The results reported in Table 1 have been obtained from ten trials and all algorithms faced a 200 seconds runtime limit.

To compare the algorithms, we only measured the time to compute minimal diagnoses, i.e. we disregarded the mapping, model conversion, as well as the time it required to communicate with the solvers. In case of `musAB` and `satAB` we parsed the execution time measured by the tools themselves, which was available in the output.

Note that for certain instances `hsdagAB`, `satAB` and `musAB` exceeded the predefined runtime threshold, which we marked with **T** in the table. Thus, for the cumulative

| | abductiveExplanations | | | hsdagAB | | | satAB | | | musAB | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Model** | **MIN** | **MAX** | **AVG** | **MIN** | **MAX** | **AVG** | **MIN** | **MAX** | **AVG** | **MIN** | **MAX** | **AVG** |
| Electrical circuit | < 1 | 129 | 19.44 | < 1 | **T** | 5131.66 | < 1 | 145.05 | 51.49 | 8 | 6881 | 700.06 |
| FCS | < 1 | 5 | 0.23 | < 1 | 18 | 1.22 | < 1 | 5.12 | 0.78 | 7 | 1974 | 419.95 |
| ACD | < 1 | 12 | 0.28 | < 1 | 3 | 0.31 | < 1 | 7 | 0.34 | 7 | 122 | 42.28 |
| Main bearing | < 1 | 1 | 0.02 | < 1 | 1 | 0.04 | < 1 | 1 | 0.07 | 11 | 269 | 93.86 |
| HIFI - FPU | < 1 | 1 | 0.04 | < 1 | 174 | 9.42 | < 1 | 6.05 | 1.98 | 7 | 469 | 141.82 |
| MiTS 1 | < 1 | 1 | 0.09 | < 1 | 1 | 0.10 | < 1 | 2.42 | 0.23 | 7 | 37 | 19.1 |
| MiTS 2 | < 1 | 12 | 0.57 | < 1 | 164891 | 3522.88 | < 1 | 11 | 2.53 | 7 | 5281 | 905.45 |
| PCB | < 1 | 1 | 0.01 | < 1 | 1 | 0.01 | < 1 | 1 | 0.12 | 7 | 12 | 9.36 |
| Inverter | < 1 | 55 | 2.62 | < 1 | **T** | 15406.82 | < 1 | **T** | 3799.10 | 8 | **T** | 14573.4 |
| Rectifier | < 1 | 4 | 0.32 | < 1 | 25830 | 233.51 | < 1 | 11450.6 | 455.11 | 8 | **T** | 17173.03 |
| Transformer | < 1 | 1 | 0.01 | < 1 | < 1 | < 1 | < 1 | 0.73 | 0.04 | 7 | 36 | 19.63 |
| Backup components | < 1 | 25 | 2.03 | < 1 | **T** | 4113.47 | < 1 | 35.84 | 9.69 | 8 | **T** | 14526.67 |

Table 1: Experimental results of the four algorithms on the experiment instances. Models, where an algorithm exceed the given run time threshold at least once, are marked with **T**.

| | Structure | | | # Diagnoses | | | | |
|---|---|---|---|---|---|---|---|---|
| **Model** | **Hyp** | **Obs** | **Th** | **Max** | **Avg** | **SF** | **DF** | **TF** |
| Electrical circuit | 32 | 17 | 52 | 792 | 189.1 | 3 | 3 | 12 |
| FCS | 17 | 17 | 56 | 28 | 3.68 | 5 | 3 | 15 |
| ACD | 13 | 16 | 52 | 12 | 2.46 | 3 | 4 | 4 |
| Main bearing | 3 | 5 | 20 | 3 | 2.34 | 3 | 0 | 0 |
| HIFI-FPU | 17 | 11 | 35 | 42 | 9.44 | 7 | 21 | 7 |
| MiTS 1 | 17 | 21 | 47 | 12 | 5.04 | 3 | 3 | 4 |
| MiTS 2 | 22 | 15 | 48 | 288 | 33.46 | 4 | 12 | 6 |
| PCB | 10 | 11 | 24 | 2 | 1.52 | 1 | 2 | 2 |
| Inverter | 29 | 38 | 165 | 200 | 21.79 | 2 | 14 | 16 |
| Rectifier | 20 | 17 | 93 | 64 | 8.1 | 16 | 32 | 64 |
| Transformer | 4 | 8 | 22 | 2 | 1.1 | 2 | 2 | 2 |
| Backup components | 25 | 30 | 114 | 252 | 19.86 | 7 | 18 | 27 |

Table 2: Features of the models and the evaluation examples. *SF*, *DF*, and *TF* refer to single, double, and triple faults, respectively.

runtimes, shown in Figure 1, we utilized the maximum of 200 seconds in cases the limit was surpassed.

Whereas some of the small runtimes are arguable due to the measurement in the milliseconds range, Table 1 as well as Figure 1 reveal that `abductiveExplanations` (Mean = 2.41 ms, SD = 12.36 ms, Median = 0 ms) outperforms `hsdagAB` (Mean = 12261.77 ms, SD = 3162.5 ms, Median = 1 ms), `satAB` (Mean = 1741.39 ms, SD = 15633.06 ms, Median = 1 ms), and `musAB` (Mean = 45947.85 ms, SD = 82289.36 ms, Median = 118 ms). Unsurprisingly, the larger considered examples are more computationally demanding, especially with the model of the electrical circuit featuring a larger set of possible hypotheses and diagnoses.

In cases where the maximum cardinality of the diagnoses is limited, HS-DAG computes solutions rather efficiently. However, in our examples, we enumerated all solutions, thus neither the size nor the number of hitting sets was restricted, which can result in some cases in an extensive graph.

The MCS-based approach performs rather poorly on the example of the converter. According to Marques-Silva *et al.* [2013] the number of SAT calls for the CLD approach depends on the size of the underlying formula, which in our case is determined by the size of the theory and the number of hypotheses, which explains the computation time for the inverter example. It is worth mentioning that in the majority of cases the hitting set computation accounted for a negligible fraction of the total runtime of `satAB`.

The performance of MARCO is very much dependent on the traversal of the graph towards a "low point" or "high point" in the power-set lattice, i.e. MUS or MSS, respectively. Thus, the number of clauses, which shape the power lattice, influences the MARCO's runtime. Therefore, in particular the larger instances require more computation time.

Note that we did not focus on an efficient encoding or any kind of pre-compilation to speed up the reasoning process. Further, in the case of MUS- and MCS-based algorithms, there is no focus on the abducibles, as for the ATMS and the HS-DAG. Thus, a large number of sets is generated, which are not of interest for the diagnostic task.

## 5 Conclusion

Abductive reasoning is of special interest in the context of diagnosis. In this paper we focused on the model-based approach utilizing a logic system description. We reviewed four different algorithms to compute abductive explanations for a propositional diagnosis problem. On the one hand, we investigated a direct strategy based on an ATMS, and on the other hand examined three approaches relying on conflict computation of an unsatisfiable model. In our tests, the conflict-based methods did not offer advantages against the ATMS. The SAT-based approaches have the drawback of not being focused on the set of abducibles, but rather enumerate all sets regardless if the clause corresponds to a hypothesis or not. Further, we could observe that in fact MCS enumeration and subsequent hitting set computation is preferable to the direct MUS approach. Surprisingly, HS-DAG did not perform well even on the smaller examples. We explain this, by the encoding of the problem, which has not been ideal for utilized theorem prover.

## Acknowledgments

The work presented in this paper has been supported by the FFG project Applied Model Based Reasoning (AMOR) under grant 842407. We would further like to express our gratitude to our industrial partner, Uptime Engineering GmbH.

## References


Elazar Birnbaum and Eliezer L Lozinskii. Consistent subsets of inconsistent systems: structure and behaviour. *Journal of Experimental & Theoretical Artificial Intelligence*, 15(1):25–46, 2003.

Tom Bylander, Dean Allemang, Michael C Tanner, and John R Josephson. The computational complexity of abduction. *Artificial intelligence*, 49(1):25–60, 1991.

Chin-Liang Chang and Richard Char-Tung Lee. *Symbolic logic and mechanical theorem proving*. Academic press, 1973.

Luca Console, Daniele Theseider Dupre, and Pietro Torasso. On the Relationship Between Abduction and Deduction. *Journal of Logic and Computation*, 1(5):661–690, 1991.

Johan De Kleer. An assumption-based TMS. *Artificial intelligence*, 28(2):127–162, 1986.

Johan de Kleer. Problem solving with the ATMS. *Artificial Intelligence*, 28(2):197–224, 1986.

Marc Denecker and Danny De Schreye. SLDNFA: an abductive procedure for abductive logic programs. *The journal of logic programming*, 34(2):111–167, 1998.

Marc Denecker and Antonis Kakas. Abduction in logic programming. In *Computational Logic: Logic Programming and Beyond*, pages 402–436. Springer, 2002.

Thomas Eiter and Georg Gottlob. The complexity of logic-based abduction. *Journal of the ACM (JACM)*, 42(1):3–42, 1995.

Gerhard Friedrich, Georg Gottlob, and Wolfgang Nejdl. Hypothesis classification, abductive diagnosis and therapy. In *Expert Systems in Engineering Principles and Applications*, pages 69–78. Springer, 1990.

Michael R Genesereth. The use of design descriptions in automated diagnosis. *Artificial Intelligence*, 24(1):411–436, 1984.

Russell Greiner, Barbara A Smith, and Ralph W Wilkerson. A correction to the algorithm in Reiter's theory of diagnosis. *Artificial Intelligence*, 41(1):79–88, 1989.

Ulrich Junker. QuickXplain: preferred explanations and relaxations for over-constrained problems. In *AAAI*, volume 4, pages 167–172, 2004.

Antonis C. Kakas, Robert A. Kowalski, and Francesca Toni. Abductive logic programming. *Journal of logic and computation*, 2(6):719–770, 1992.

Roxane Koitz and Franz Wotawa. On the computational feasibility of abductive diagnosis for practical applications. In *Proceedings of the 9th IFAC Symposium, SAFEPROCESS 2015*, 2015. To appear.

Roxane Koitz and Franz Wotawa. Sat-based abductive diagnosis. In *DX-15, International Workshop on the Principles of Diagnosis*, 2015. To appear.

Hector J Levesque. A knowledge-level account of abduction. In *IJCAI*, pages 1061–1067, 1989.

Mark H Liffiton and Karem A Sakallah. Algorithms for computing minimal unsatisfiable subsets of constraints. *Journal of Automated Reasoning*, 40(1):1–33, 2008.

Mark H Liffiton, Alessandro Previti, Ammar Malik, and Joao Marques-Silva. Fast, flexible MUS enumeration. *Constraints*, pages 1–28, 2015.

Li Lin and Yunfei Jiang. The computation of hitting sets: review and new algorithms. *Information Processing Letters*, 86(4):177–184, 2003.

Joao Marques-Silva, Federico Heras, Mikolás Janota, Alessandro Previti, and Anton Belov. On computing minimal correction subsets. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 615–622. AAAI Press, 2013.

Pierre Marquis. Consequence finding algorithms. In *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, pages 41–145. Springer, 2000.

Sheila A McIlraith. Logic-based abductive inference. *Knowledge Systems Laboratory, Technical Report KSL-98-19*, 1998.

Michel Minoux. LTUR: A simplified linear-time unit resolution algorithm for horn formulae and computer implementation. *Information Processing Letters*, 29(1):1–12, 1988.

Gustav Nordh and Bruno Zanuttini. What makes propositional abduction tractable. *Artificial Intelligence*, 172(10):1245–1284, 2008.

Ekaterina Ovchinnikova, Niloofar Montazeri, Theodore Alexandrov, Jerry R Hobbs, Michael C McCord, and Rutu Mulkar-Mehta. Abductive reasoning with a large knowledge base for discourse processing. In *Computing Meaning*, pages 107–127. Springer, 2014.

Bernhard Peischl and Franz Wotawa. Computing diagnosis efficiently: A fast theorem prover for propositional horn theories. In *Proc. of the 14th Int. Workshop on Principles of Diagnosis*, pages 175–180, 2003.

Ingo Pill, Thomas Quaritsch, and Franz Wotawa. From conflicts to diagnoses: An empirical evaluation of minimal hitting set algorithms. In *22nd Int. Workshop on the Principles of Diagnosis*, pages 203–210, 2011.

David Poole and Keiji Kanazawa. A decision-theoretic abductive basis for planning. In *AAAI Spr. Symp. on Decision-Theoretic Planning*, 1994.

David Poole, Randy Goebel, and Romas Aleliunas. *Theorist: A logical reasoning system for defaults and diagnosis*. Springer, 1987.

Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.

Bert Van Nuffelen. A-system: Problem solving through abduction. *BNAIC'01 Sponsors*, 1:591–596, 2001.

Franz Wotawa, Ignasi Rodriguez-Roda, and Joaquim Comas. Abductive Reasoning in Environmental Decision Support Systems. In *AIAI Workshops*, pages 270–279, 2009.

Franz Wotawa. Failure mode and effect analysis for abductive diagnosis. In *Proceedings of the International Workshop on Defeasible and Ampliative Reasoning (DARe-14)*, volume 1212. CEUR Workshop Proceedings, ISSN 1613-0073, 2014. http://ceur-ws.org/Vol-1212/.