Applied Intelligence (2020) 50:1558–1572
https://doi.org/10.1007/s10489-019-01575-5

Faster horn diagnosis - a performance comparison of abductive reasoning algorithms

Roxane Koitz-Hristov[1] · **Franz Wotawa**[1]

Published online: 28 January 2020

Abstract

Abductive inference derives explanations for encountered anomalies and thus embodies a natural approach for diagnostic reasoning. Yet its computational complexity, which is inherent to the expressiveness of the underlying theory, remains a disadvantage. Even when restricting the representation to Horn formulae the problem is NP-complete. Hence, finding procedures that can efficiently solve abductive diagnosis problems is of particular interest from a research as well as practical point of view. In this paper, we aim at providing guidance on choosing an algorithm or tool when confronted with the issue of computing explanations in propositional logic-based abduction. Our focus lies on Horn representations, which provide a suitable language to describe most diagnostic scenarios. We illustrate abduction via two contrasting problem formulations: direct proof methods and conflict-driven techniques. While the former is based on determining logical consequences, the later searches for suitable refutations involving possible causes. To reveal runtime performance trends we conducted a case study, in which we compared publicly available general purpose tools, established Horn reasoning engines, as well as new variations of known methods as a means for abduction.

Keywords Abductive reasoning · Model-based diagnosis · Abductive diagnosis

1 Introduction

Within the last decades, an extensive number of approaches and tools for abductive reasoning have been developed within the Artificial Intelligence community. These methods are tailored to various underlying artifacts and diverse domains, such as plan recognition [4], test case generation [36], ontology debugging [50], or human behavior interpretation [19]. Still, the most prominent application area of abduction is diagnosis, i.e., identifying root causes for a given set of observed anomalies. Model-based diagnosis (MBD) has been proposed as a means to locate faults by reasoning directly on a description of the system [45]. While the traditional consistency-based version of MBD extracts conflicts from discrepancies between predicted and observed behavior and subsequently derives diagnoses, the abductive variant is founded in logic-based abduction [34]. Logic-based abduction is defined as the search for a set of consistent abducible propositions that together with a background theory entail the observations. Additional restrictions, such as minimality, are often placed on the solutions. Originally, abductive model-based approaches operate on formalizations of the faulty system behavior. Such failure knowledge can usually be expressed as Horn theories [6]. Besides being a suitable representation for diagnostic models, the complexity results for this subset of logic are less daunting; that is, abduction for general theories is located on the second level in the polynomial hierarchy, while for Horn models the complexity is lowered by one level [10].

In the following example, we describe a simple diagnosis problem taken from the industrial wind turbine domain. Gearbox lubrication is an essential aspect in regard to the reliability of wind turbines as it protects the contact surfaces of gears and bearings from excessive wear and prevents overheating. In a simplified scenario, the insufficient lubrication can be caused by various component failures. On the one hand, a broken oil pump leads

This work has been previously published as part of the PhD thesis [30].

✉ Roxane Koitz-Hristov
rkoitz@ist.tugraz.at

Franz Wotawa
wotawa@ist.tugraz.at

[1] Insitute for Software Technology, Graz University of Technology, Inffeldgasse 16b/II, Graz 8010, Austria

to reduced oil pressure. This decreased pressure diminishes the flow rate of oil through the system. On the other hand, damages to the oil cooler eventually cause an overheating of the oil, which in conjunction with a blocked oil filter also negatively affects the lubrication resulting from a reduction in the film thickness at bearing and gear contacts. These relations can be described by a set of Horn clauses: $HC = \{Damaged\_pump \rightarrow reduced\_pressure, reduced\_pressure \rightarrow poor\_lubrication, Broken\_filter \wedge overheating \rightarrow poor\_lubrication, Cooler\_leaks \wedge Cooler\_cracks \rightarrow overheating\}$. Given the interactions between component faults and their consequences, we can identify two root causes of insufficient lubrication: (1) a damaged oil pump and (2) a broken filter in conjunction with cracks and leaks of the oil cooling component. These causes constitute the faults we want to identify during diagnosis. In the remainder of this paper, we make use of this running example to illustrate the different abductive reasoning approaches.

As abductive inference can be applied to different tasks, various techniques for solving logic-based abduction problems have emerged. Some frame the issue of deriving explanations as the search for logical consequences, i.e., diagnoses are derived deductively. Inoue [22], for example, proposes Skipping Ordered Linear (SOL) tableau calculus for extracting consequences of interest equivalent to abductive explanations. Another consequence finding method is kernel resolution [49]. Similar to SOL-resolution, kernel resolution focuses on computing consequences of clauses of interest. This technique can be implemented efficiently with Zero-Suppressed Binary Decision Diagrams and performs well on large problem instances. Another classical technique regarding diagnosis is the Assumption-based Truth Maintenance System (ATMS) [7]. The ATMS has been used extensively in the context of consistency-based diagnosis, but given the information it records the ATMS can further generate abductive explanations by deriving minimal logical consequences. Abductive Logic Programming (ALP) [24] has been introduced as a means to declaratively solve abduction problems within the framework of logic programming. Logic programming is augmented with abducible predicates, i.e., ground instances that may be part of a diagnosis, and integrity constraints posing restrictions on what constitutes an admissible solution. Besides specific tools such as the $\mathcal{A}$-System [25], ALP can be realized using Answer Set Programming (ASP) [47].

A different framework for abductive diagnosis is proof-tree completion-style abduction. Here the diagnosis problem is rewritten in an entailment preserving way, such that explanations are equivalent to conflicts [37]. Using this notion of abduction, innovations in extracting Minimal Unsatisfiable Subsets (MUSes) of logical formulae, which are equivalent to conflicts, can be exploited to obtain abductive diagnoses [32]. A common approach to MUSes extraction is to first compute their hitting set duals Minimal Correction Subsets (MCSes) and subsequently derive the irreducible hitting sets [33]. Recently, inspired by the success of implicit hitting set algorithms for MaxSAT, Saikko et al. [46] have introduced a method that computes one minimum-cost abductive explanation by utilizing Integer Programming to iteratively derive hitting sets. Each hitting set is then checked whether it constitutes a solution. Ignatiev, Morgado, and Marques-Silva [21] report on an improvement of this method by deriving the hitting sets using a MaxSAT solver.

Yet contemplating the collection of approaches and tools for abduction, the question, which strategy to follow, remains. An analysis seeking to answer this question for consistency-based diagnosis algorithms exists [40]. The authors compare two strategies to derive diagnoses: on the one hand methods that directly compute the solutions given the system description and symptoms and on the other hand conflict-based techniques that exploit contradictions arising from correct behavior assumptions and the observations. Feldman et al. [13] introduce a benchmark framework for executing diagnosis methods for physical systems under identical conditions. The performance evaluation presented includes rule-based, (consistency) model-based, data driven, and stochastic fault detection and identification approaches.

While there is research comparing the consensus among different abduction formalizations [15, 37], assessing an emerging tool's performance in regard to previously proposed mechanisms [9, 46], or applying a reasoning system to different domains [39], a broader empirical evaluation as available in the consistency-based case is missing for abductive Horn diagnosis. For instance, McIlraith [37] reviews different formal characterizations, frameworks, techniques, and application areas for abductive inference. While the comparison is broad and extensive, it is only theoretic in nature. Egly et al. [9] describe a translation of different non-monotonic reasoning tasks, such as abduction, into the evaluation problem of quantified boolean formulas and compare an implementation of their method to non-monotonic reasoning solvers, such as dlv [12] or Theorist [44], on multiple benchmark problems. The focus of this work is to show whether non-monotonic reasoning tasks can be solved efficiently by translation to quantified boolean formulas; hence, only a portion of the assessment concerns abduction. Ng and Mooney [39] have conducted an empirical analysis comparing the general abduction system ACCEL to a set of problems in plan recognition as well as consistency model-based and set-covering [43] diagnosis. While the authors could show that a general solver can derive explanations within the two application domains in reasonable time, their experiments consider solely a single tool.

Hence in this paper, we aim at providing guidance for deciding on a procedure focusing on propositional Horn abduction. Our work is based on previous research. On the one hand, we have assessed abductive inference on models restricted to bijunctive Horn clauses [27, 28] and on the other hand, we have developed a meta-approach that has been evaluated in regard to different abduction algorithms [31]. In this paper, we compare the performance of propositional Horn clause abduction within two general directions, i.e., direct and conflict-driven techniques, similar to the work by Nica et al. [40] in the consistency-based case. To identify runtime trends, we conducted an empirical evaluation based on publicly available tools, implementations of various traditional abductive reasoning algorithms, and adaptations of recent and proven techniques. The diagnosis problems utilized within our analysis are taken on the one hand from real-world domains and on the other hand are artificially created similar to practical fault identification scenarios. One of our goals is to discover whether a certain approach is dominant on these reasonable samples. Besides seeking information on efficiency differences between direct proof and conflict-driven algorithms, we aim at showing to what extend more general off-the-shelf tools can compete with specific Horn reasoning methods.

2 Preliminaries

Abductive inference allows us to derive possible explanations for a set of given observations. In logic-based abduction, we rely on the notion of entailment; a set of premises ψ logically entails a conclusion ϕ if and only if for any interpretation in which ψ holds ϕ is also true. We write this relation as $\psi \models \phi$ and call ϕ a logical consequence of ψ. An abductive explanation or diagnosis is composed of a set of abducible propositions, i.e., fragments permitted to be part of a solution, entailing the observed symptoms together with the theory while not leading to a contradiction. In the context of fault identification, these abducible propositions are usually abnormality assumptions about components, while in the medical domain diseases represent the variables allowed to form a diagnosis.

The complexity of logic-based abduction is connected to the characteristics of the underlying model [2, 41]. For general propositional theories or models in clausal form abductive reasoning is located in the second level of the polynomial hierarchy, while for Horn theories the problem is NP-complete [10, 17]. A Horn clause is defined as a disjunction of literals featuring at most one positive literal and can be represented by a rule, i.e., $\neg a_1 \vee \ldots \vee \neg a_n \vee a_{n+1}$ is equivalent to $a_1 \wedge \ldots \wedge a_n \rightarrow a_{n+1}$. Many diagnostic reasoning engines restrict the model to Horn clause sentences, which are usually expressive enough to convey the necessary relations.

Due to the complexity results and suitability of Horn representation in the context of diagnosis, we define a Propositional Horn Clause Abduction Problem (PHCAP) similarly to Friedrich, Gottlob, and Nejdl [17]. The PHCAP relies on a formalization of the system encoded within a propositional Horn theory *Th* over a finite set of propositional variables *A*. Moreover, we have to state which primitives can be abduced, i.e., variables allowed to be part of an explanation. We refer to these abducible propositions as hypotheses or assumptions. The set *Hyp* contains all hypotheses and a model or knowledge base is formally defined as a tuple *(A,Hyp,Th)*.

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.

Example 1 Let us again state the causal relations negatively affecting the gearbox lubrication in an industrial wind turbine: A damaged oil pump leads to loss of oil pressure and flow, hence the lubricant distribution through the system is decreased. As an alternative, a blockage of the filter in the oil cooling system can occur which in conjunction with overheating of the oil diminishes the lubrication of the gear and bearing surfaces. Oil overheating arises from leaks or cracks in the oil cooler. These circumstances can be easily described by a KB[1]:

$$A = \left\{ \begin{array}{c} Damaged\_pump, Broken\_filter, Cooler\_leaks, \\ Cooler\_cracks, reduced\_pressure, poor\_lubrication, overheating \end{array} \right\}$$

$$Hyp = \left\{ Damaged\_pump, Broken\_filter, Cooler\_leaks, Cooler\_cracks \right\}$$

$$Th = \left\{ \begin{array}{c} Damaged\_pump \rightarrow reduced\_pressure, reduced\_pressure \rightarrow poor\_lubrication, \\ Broken\_filter \wedge overheating \rightarrow poor\_lubrication, \\ Cooler\_leaks \wedge Cooler\_cracks \rightarrow overheating \end{array} \right\}$$

[1] To distinguish assumptions from ordinary propositions, they start with a capitalized character.

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

A diagnosis problem is formed when in addition to the *KB* a set of observations is applied for which we want to derive the root causes. In our definition, observations may only be a conjunction of propositions and not an arbitrary logical sentence.

Definition 3 (Diagnosis; Solution of a PHCAP) Given a PHCAP (A,Hyp,Th,Obs). A set $\Delta \subseteq$ Hyp is a solution iff $\Delta \cup$ Th $\models$ Obs and $\Delta \cup$ Th $\not\models \bot$.

Definition 4 (Parsimonious Diagnosis) A solution Δ to a PHCAP is parsimonious or minimal iff no set $\Delta' \subset \Delta$ is a solution to the same PHCAP.

A solution to a PHCAP, i.e., a diagnosis, must be a subset of the abducibles, be consistent together with the theory and logically entail the observations. Generally, we are only interested in subset minimal explanations. Especially considering a practical application of abductive diagnosis, explanation supersets provide no additional information useful in a real-world context. We define Δ-Set as the set of all (minimal) diagnoses obtained from the PHCAP, i.e., $\Delta\text{-Set} = \{\Delta | \Delta \subseteq Hyp, \Delta \cup Th \models Obs\}$. To determine all diagnoses for a PHCAP one can test all subsets of hypotheses to determine whether they entail the observations and are consistent. This procedure, however, requires exponential runtime.

Example 2 Given the KB from the previous example, assume we observe an insufficient greasing of the gearbox, i.e., $Obs = \{poor\_lubrication\}$. There are two parsimonious explanations; either the oil pump is damaged , i.e., $\Delta_1 = \{Damaged\_pump\}$, or leaks and cracks in the cooler cause oil overheating which in combination with a filter failure leads to inadequate lubrication, i.e., $\Delta_2 = \{Broken\_filter, Cooler\_leaks, Cooler\_cracks\}$. Hence, $\Delta\text{-Set} = \{\{Damaged\_pump\}, \{Broken\_filter, Cooler\_leaks, Cooler\_cracks\}\}$.

2.1 Proof-tree completion

Considering Definition 3, we can characterize two more approaches for computing solutions to a PHCAP: proof-tree completion [37] and consequence finding [35]. In proof-tree completion, abductive diagnosis is re-framed to the search for a refutation proof consisting of abducible propositions. From the definition, we know that each explanation together with the theory has to entail the observations, i.e., $\Delta \cup Th \models Obs$. By logical equivalence, we can reformulate this relation to $\Delta \cup Th \cup \{\neg Obs\} \models \bot$, where $\{\neg Obs\}$ is the disjunction containing a negation of each observation, i.e., $\bigvee_{o_i \in Obs} \neg o_i$. Thus, we can rewrite the theory in such a way that a contradiction arises given the negated observations. To extract explanations, the derived conflicts have to be propositions from *Hyp* and again no inconsistency may arise from the diagnoses.

Definition 5 (Conflict) Given a PHCAP(A,Hyp,Th,Obs) a conflict CO is a set $\{c_1, \ldots, c_k\} \subseteq Hyp$ such that $CO \cup Th \cup \{\neg o_1 \vee \ldots \vee \neg o_n\}$ is inconsistent, where $\{o_1, \ldots, o_n\} = Obs$. A conflict is minimal iff there is no conflict CO' such that $CO' \subset CO$.

Definition 6 (Conflict-Driven Diagnosis) Given a PHCAP(A,Hyp,Th,Obs). A set Δ is a solution if Δ is a conflict and $\Delta \cup$ Th $\not\models \bot$. A solution is parsimonious, iff Δ is a minimal conflict.

Example 3 Given our PHCAP from before, we assume sufficient lubrication, i.e., $\neg poor\_lubrication$, and that all our hypotheses are true, that is, the filter and pump are behaving incorrectly and there are cracks and leakages in the cooler. This leads to conflicts: for instance, according to the theory given a faulty pump, we must observe poor lubrication; thus, $\{Damaged\_pump\} \cup Th \cup \{\neg poor\_lubrication\}$ is inconsistent. Extracting only the hypotheses we retrieve the first minimal conflict $CO_1 = \{Damaged\_pump\}$. Since the conflict is consistent with the theory *Th*, CO_1 is a minimal diagnosis. The second parsimonious conflict and diagnosis is $CO_2 = \{Broken\_filter, Cooler\_leaks, Cooler\_cracks\}$ and can be extracted analogously.

2.2 Consequence finding

By further rewriting the relation stated in Definition 3, we can construct $Th \cup \{\neg Obs\} \models \{\neg\Delta\}$, where $\{\neg\Delta\} = \bigvee_{\delta_j \in \Delta} \neg\delta_j$. In this scenario, abductive explanations are obtained in a deductive manner by finding the logical consequences of the theory and the negated observations comprising negations of hypotheses.

Definition 7 (Consequence) Given a PHCAP (A,Hyp,Th,Obs) a consequence C is a set $\{c_1, \ldots, c_k\} \subseteq Hyp$ such that $Th \cup \{\neg o_1 \vee \ldots \vee \neg o_n\} \models \{\neg c_1 \vee \ldots \vee \neg c_k\}$, where $\{o_1, \ldots, o_n\} = Obs$. A consequence is minimal iff there is no consequence C' such that $C' \subset C$.

Definition 8 (Consequence-Finding Diagnosis) Given a PHCAP (A,Hyp,Th,Obs). A set Δ is a solution iff Δ is a

consequence and $\Delta \cup \text{Th} \not\models \bot$. A solution is parsimonious, iff Δ is a minimal consequence.

Example 4 Again we state sufficient greasing, i.e., $\neg poor\_lubrication$. The pump cannot be faulty given this observation since the relation in *Th* postulates that if the pump is behaving incorrectly, then there must be inadequate lubrication. Looking at the entailment relation, we can deduce that the negation of *Damaged_pump* is a logical consequence of the theory and the observation, i.e., $Th \wedge \neg poor\_lubrication \models \neg Damaged\_pump$. Hence, {*Damaged_pump*} is one of the consequences as defined in Definition 7. Since the consequence is further consistent with *Th*, {*Damaged_pump*} is a minimal fault diagnosis. Similarly, we can derive the second diagnosis.

3 Empirical evaluation set-up

In this section, we present our empirical evaluation framework and report on the obtained results[2]. Our focus lies on identifying runtime trends suggesting the advantage of one approach over the other. Further, we want to contribute to answering the questions (1) whether a general-purpose solver can be used for an efficient diagnosis in practice or if a specifically tailored engine is required for achieving convincing performance and (2) whether there is a superiority of direct or conflict driven methods.

First, we give some insights into the implementations of the abduction methods used. Then, we report on the benchmarks considered in the experiments. Particularly, we utilized artificially generated diagnosis problems as well as real world failure models. Subsequently, we discuss the results of the experiments all obtained from a Mac Pro (Late 2013) with a 2.7 GHz 12-Core Intel Xeon ES processor and 64GB of RAM running OS X 10.10.5.

3.1 Algorithms

In previous work [31], we have described several diagnosis algorithms which we implemented in Java as well as exploited available tools realizing the abductive reasoning procedures. The following implementations are used in the empirical evaluation:

- **Abduction with the ATMS** (*ATMS*): Within our evaluation we exploit a Java implementation of ATMSXPLAIN based on a Java implementation of an ATMS in its original form, i.e., restricted to Horn theories. Since we are dealing with Horn clauses, we utilize LTUR as the theorem prover allowing us to compute inferences efficiently on our models. In particular, we exploit a Java implementation of an assumption-based LTUR of the diagnosis engine jdiagengine[3] [42]. We refer to this entire abduction procedure simply as *ATMS* within our experiments.
- **Abduction as Consequence Finding via SOL-resolution** (*CF*): We created a consequence finding set-up *CF* using a Java implementation of CONSQXPLAIN. To enumerate the characteristic clauses we employ SOLAR[4] [38], which is a Java implementation of SOL-tableaux calculus for first order full clausal theories. The tool provides various pruning methods, which ensure, for instance, that redundant tableaux are not generated. We invoked SOLAR with the default setting, which executes a depth-first search within the tableaux repeatedly, incrementing the depth limit for each iteration.
- **Conflict-Driven Search via HS-DAG**: To realize HS-DAGXPLAIN we use jdiagengine [42], which implements a conflict-driven search via HS-DAG coupled with an incremental assumption-based LTUR. LTUR does not guarantee to return minimal conflicts, hence the contradicting assumptions have to be reduced to derive minimal explanations. We use two set-ups:
  - *HS-DAG*: For this procedure, we simply record all refutations returned within the search on the HS-DAG. After the computation has finished, all conflict supersets are removed resulting in parsimonious diagnoses.
  - *HS-DAG$_{QX}$*: We adapted jdiagengine's implementation to minimize refutations right away after being returned from LTUR. In particular, we created a Java implementation of QuickXplain to reduce each conflict to a parsimonious one. Our version of QuickXplain is based on assumptions rather than on constraints with preferences exploits the assumption-based LTUR for its consistency checks. Every refutation minimized by QuickXplain already constitutes an abductive explanation.
- **Conflict-Driven Search via Power Set Exploration**: We realized XPLORER in Java. To favor unsatisfiable cores early on in the computation, we implemented Arif et al.'s [1] method to find maximal model seeds. In

[2] An executable for our evaluation and the benchmarks are available on https://www.dropbox.com/sh/c6ykbm478ixmqhu/AAD9yKYtHtTgN-YsbV5fN10_a?dl=0.

[3] https://www.ist.tugraz.at/modremas/index.html

[4] Used version: SOLAR 2 (Build 315)

Table 1 Sample set statistics artificial examples

	Artificial samples I				Artificial samples II			
	MIN	MAX	AVG	MED	MIN	MAX	AVG	MED
$\|Hyp\|$	10	504	275.07	320.00	12	235	120.42	112.50
$\|A \setminus Hyp\|$	6	6,466	1,903.23	1,668.00	13	1,055	252.74	180.00
$\|Th\|$	10	7,186	2,950.10	2,731.00	20	1,146	416.70	358.50
$\|Obs\|$	1	5	2.86	3.00	1	5	2.72	2.50
$\|\Delta$-Set$\|$	1	50	2.76	1.00	1	58,520	500.71	2.00

addition, we used the SAT solver SAT4J[5] [8] in order to determine the satisfiability of the Boolean formula representing the map. To shrink a found unsatisfiable seed to an MUS, we use two extraction mechanisms:

  - *XPLorer*: For our first set-up we implemented the insertion-based MUS extraction algorithm as suggested by Arif et al. [1] using jdiagengine's incremental assumption-based LTUR.
  - $XPLorer_{QX}$: Since each unsatisfiable seed already constitutes a conflict and the MUS extraction merely reduces it towards a minimal contradiction, we again applied the assumption-based QuickXplain implementation to minimize the unsatisfiable seed to an MUS, i.e., abductive explanation.

- **Abduction under Stable Model Semantics** (*ASP*): By exploiting an encoding of propositional abduction by Saikko et al. [46][6] we compute diagnoses via the state-of-the-art C++ ASP solver clingo 4.5.4[7] [18]. The encoding generates minimum-cost solutions using the negation of the conjunction of observations based on a technique called saturation to determine whether the entailment relation between diagnoses and manifestations exists [5, 11]. As we want to enumerate all subset minimal answer sets, we remove the optimization criteria included in the encoding and apply domain-specific heuristics to the solver over the command line[8]. We call this approach *ASP* within the evaluation.

In *CF* and *ASP*, we invoke both general-purpose solvers using a separate process inside our implementation, even though we could use SOLAR directly in our Java code. We use this set-up to mimic that both tools are independent from the used programming language of the abduction procedure and simply function as black boxes returning the diagnoses.

3.2 Data

To asses the various abductive reasoning mechanisms presented, we chose to use two different types of models. On the one hand, we created artificial Horn clause models, which are similar to diagnosis knowledge used in practice. On the other hand, we exploit knowledge on how component-based failures affect real world systems. The information is automatically compiled into a Horn theory via a mapping function and subsequently used in the context of abductive MBD.

3.2.1 Artificial benchmarks

By means of a sample generator, we constructed artificial PHCAPs [31]. The rules contained within the theory are based on several parameters; n is the minimum number of literals in a rule's body. Those literals are chosen randomly from A. Thus, this value indirectly determines the number of hypotheses present within the entire model. r bounds the number of clauses generated per body, while o limits the overlap between head literals. The head of each rule can only contain elements from $A \setminus Hyp$, thus, we do not have assumptions that are caused by another hypothesis. Further, the fabricated models do not contain any cycles. The parameter k determines the maximum number of manifestations to be explained. Those observations are randomly selected from $A \setminus Hyp$ and for simplicity comprise only positive propositions.

The examples constructed by the generator are similar to real world samples. Just consider, for instance, medical diagnosis, where the knowledge mainly captures how a set of assumptions about diseases affects a set of symptoms. Usually, we cannot observe cycles or an excessive number of implication levels within this type of knowledge.

For the empirical evaluation we fabricated two different sets of artificial samples; *Artificial Samples I* was constructed with k=5, r=o=15 and n=1 and contains 166 PHCAPs. For *Artificial Samples II* the example generator

[5] https://www.sat4j.org/

[6] www.cs.helsinki.fi/group/coreo/abhs/

[7] www.potassco.org/clingo/

[8] clingo is invoked with `--heuristic=Domain`, `--enum-mod=domRec`, and `--dom-mod=5,16`. This ensures the computation of subset minimal answer sets.

Table 2 Sample set statistics FMEA samples

	FMEA Samples			
	MIN	MAX	AVG	MED
$\|Hyp\|$	3	90	26.16	20.00
$\|A \setminus Hyp\|$	5	83	26.60	21.00
$\|Th\|$	12	298	70.59	37.00
$\|Obs\|$	1	29	10.79	10.00
$\|\Delta$-Set$\|$	1	2,288	67.98	6.00

was invoked with $k{=}o{=}r{=}5$ and $n{=}1$ and created 118 diagnosis problems. Table 1 summarizes the statistics for the generated Horn models.

3.2.2 Real world samples

To make use of abductive inference in industrial applications, Wotawa [51] proposes to automatically extract the required system descriptions from failure assessments which are commonly used in practice. In particular, Failure Mode and Effects Analysis (FMEA) provides all information necessary to function as a basis of an abductive diagnosis model. FMEA as a reliability analysis tool is growing in importance as it has been established as a mandatory task in certain industries, especially for systems that require a detailed safety assessment [3]. An FMEA provides a systematic analysis of possible component faults and the consequences said faults have on the system behavior and function [20].

For our experiments, we obtained several publicly available as well as project internal FMEAs comprising diverse technical systems and sub-systems. Overall we used twelve FMEAs for the evaluation, which cover for instance electrical circuits, a connector system by Ford, printed circuit boards, as well as components, such as rectifier, inverter, transformer, main bearing of an industrial wind turbine. Subsequently, we created for each analysis a corresponding abductive knowledge base via the mapping described by Wotawa [51]. The theories resulting from the transformation are characterized by a set of bijunctive definite Horn clauses, i.e., each clause features a single hypothesis in the body and a proposition from $A \setminus Hyp$ as head. Based on the models we created 213 diagnosis problems, where we randomly chose observations from the set of effects described in the assessment. The statistical information on the FMEA models is shown in Table 2.

4 Results

As mentioned, all algorithms and tools are implemented in Java, except clingo, which is a C++ ASP solver. Each method was invoked ten times on each PHCAP; for instance, we collected 1,660 data points per abductive reasoning approach in the context of *Artificial Samples I*. Since we are interested in runtime trends and trade-offs between the algorithms, our evaluation focuses on computation times. Note here that we record the time to extract all minimal explanations. Although we could use some of the approaches to derive a partial set of solutions, we are not considering this possibility in the evaluation. In addition, we disregarded the rewriting of the model, the creation of the different input formats as well as the time it required to communicate with the solvers. In case of SOLAR and clingo we parsed the execution times recorded by the tools themselves which were available in the output[9].

Table 3 gives an overview of the evaluation results on all sample sets for our seven abduction methods. For clarity, we put the best values per category in bold face. Each computation faced a runtime limit of twenty minutes. In the rows categorized *Runtimes* we provide the performance statistics based on the samples that were computed in time, i.e., all executions where the approach exceeded the limit are not considered within these numbers. The number of samples completed in time is reported in Table 3 in row **# solved**. For example, for *Artificial Samples I* only *ATMS* managed to derive all solutions for all executions within the given time frame. Additionally, we report the number of samples an approach has computed the diagnoses the fastest in row **# fastest**. In contrast to *Runtimes*, $Runtimes_\theta$ contains results where each timed out execution is penalized with $\theta = 40 minutes$[10]. We do not report on the minimal result values in $Runtimes_\theta$, as they are equivalent to *Runtimes*, nor on the maximum execution times which are either the same as for *Runtimes* or the penalized runtime.

4.1 Artificial benchmarks

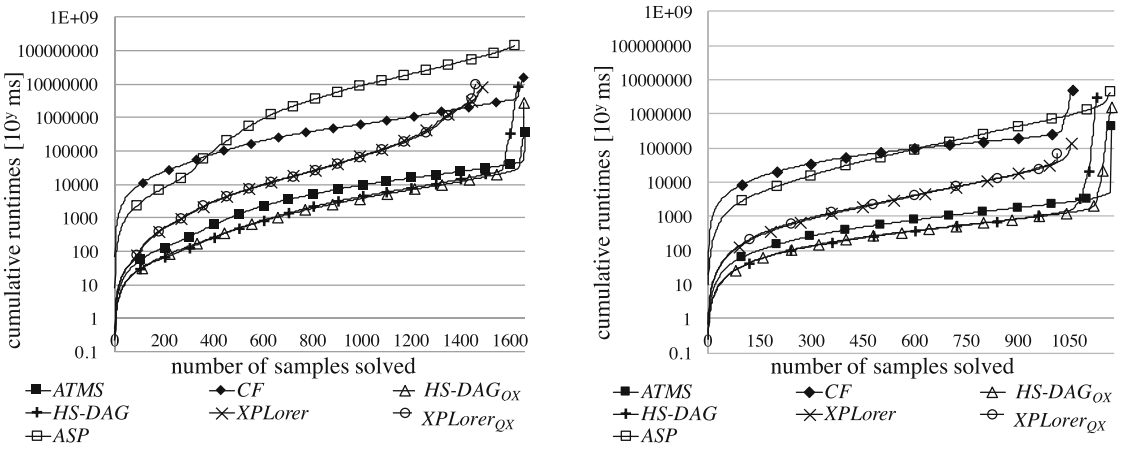
To illustrate the runtimes of the algorithms, we ordered all successful sample runs according to their execution time. In Fig. 1 a and b we report on the number of samples solved

[9] Since their runtime performance was measured in seconds, we converted the values to milliseconds for our analysis.

[10] We set θ to twice the runtime threshold value. Of course, this number is somewhat arbitrary, since we do not know the real computation time of an approach exceeding the time limit.

Table 3 Runtime results

			ATMS	CF	HS-DAG	HS-DAG$_{QX}$	XPLorer	XPLorer$_{QX}$	ASP
Artificial Samples I [1,660 runs]	*Runtimes [in ms]*	**# solved**	**1,660**	1,650	1,629	1,654	1,485	1,457	1,618
		# fastest	183	0	**820**	629	15	13	0
		MIN	0.34	70.00	0.23	**0.22**	0.49	0.35	8.00
		MAX	**170,838.66**	899,370.00	1,177,876.77	952,760.53	567,187.64	904,397.79	1,156,071.00
		AVG	**226.28**	9,387.55	5,397.07	1,600.36	5,466.45	6,590.33	88,315.45
		MED	19.91	1,080.00	9.20	**8.04**	88.96	80.22	17,931.50
		SD	**4,529.92**	65,366.46	52,305.30	34,443.66	27,536.54	48,291.29	174,211.35
	Runtimes$_\theta$ [in ms]	AVG	**226.28**	23,788.83	50,115.55	10,269.27	257,902.21	299,278.38	146,803.85
		MED	19.91	1,110.00	9.76	**8.13**	139.55	130.70	20,678.50
		SD	**4,529.92**	196,183.91	328,376.83	148,023.42	736,034.18	785,667.50	401,804.10
Artificial Samples II [1,180 runs]	Runtimes [in ms]	**# solved**	**1,171**	1,139	1,130	1,170	1,055	1,015	1,168
		# fastest	231	0	**534**	415	0	0	0
		MIN	0.39	80.00	0.26	**0.25**	0.69	0.72	10.00
		MAX	425,393.84	389,900.00	391,309.67	217,275.60	27,919.00	**27,649.77**	315,250.00
		AVG	367.55	4,941.55	2,783.68	1,376.12	132.65	**67.36**	4,031.42
		MED	2.23	220.00	**0.99**	1.00	14.13	12.88	470.00
		SD	12,431.07	33,547.10	27,328.01	12,864.20	1,048.81	**870.91**	22,215.67
	Runtimes$_\theta$ [in ms]	**AVG**	**9,517.29**	46,464.77	53,513.19	11,533.95	127,237.24	167,854.55	16,193.81
		MED	2.26	240.00	1.02	**1.00**	20.45	19.49	490.00
		SD	**105,143.70**	221,759.43	242,745.11	110,664.17	369,418.07	416,327.19	122,059.98
FMEA Samples [2,130 runs]	Runtimes [in ms]	**# solved**	2,119	**2,130**	1,758	2,020	1,918	1,650	2,020
		# fastest	**1,267**	19	294	539	8	3	0
		MIN	0.33	60.00	**0.24**	0.25	0.56	0.56	6.00
		MAX	916,524.17	375,020.00	1,191,864.18	140,108.57	883,309.51	903,439.88	**14,680.00**
		AVG	1,854.24	2,375.50	22,709.35	641.59	10,685.16	5,662.87	**239.74**
		MED	**0.90**	200.00	1.87	1.26	23.36	18.05	32.00
		SD	30,796.96	24,030.25	111,354.35	6,775.37	68,640.88	47,594.18	**1,299.25**
	Runtimes$_\theta$ [in ms]	**AVG**	14,239.04	**2,375.50**	437,898.14	124,552.12	248,494.90	545,231.80	124,171.03
		MED	**0.91**	200.00	4.25	1.39	29.77	30.49	35.00
		SD	174,655.60	**24,030.25**	908,437.40	531,157.15	718,422.99	1,001,498.81	531,206.63



(a) *Artificial Samples I* (b) *Artificial Samples II*

Fig. 1 Numbers of diagnosis samples solved over time for the artificially generated diagnosis problems

for growing cumulative log runtime for *Artificial Samples I* and *Artificial Samples II*, respectively.

From the graphics in Fig. 1 we can deduce that all algorithms experience an exponential runtime curve. In addition, the plots show that the same technique with different minimization algorithms, i.e., *XPLorer* and $XPLorer_{QX}$ as well as *HS-DAG* and $HS\text{-}DAG_{QX}$, feature a similar performance and only diverge in regard to their efficiency towards the more runtime expensive instances.

ATMS is capable of solving the most PHCAPs within the given time limit for both sample sets, followed by $HS\text{-}DAG_{QX}$. In contrast, approaches based on the exploration

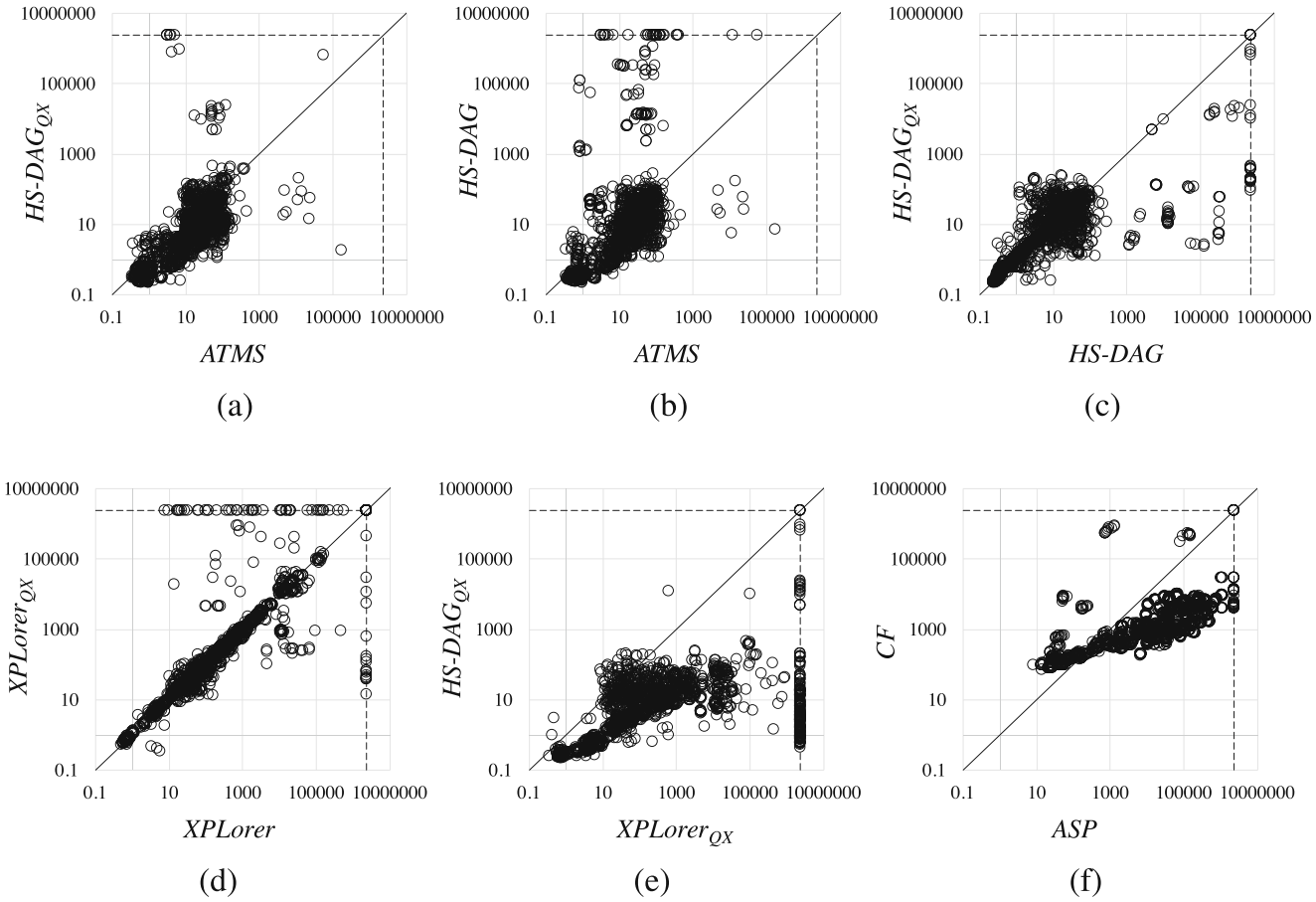


Fig. 2 Scatter plots comparing the penalized runtimes [10^y ms] on *Artificial Samples I*

of the power set fail to compute solutions for up to thirteen percent of the examples in time. Hence, the execution times in *Runtimes* in Table 3 can be deceiving since they, for instance, might convey that $XPLorer_{QX}$ provides preferable results in regard to the maximum and average runtime. Yet considering the penalized results, we can see that the approach is not competitive. From the penalized runtimes, we deduce that *ATMS* and $HS\text{-}DAG_{QX}$ are superior in comparison to the remaining techniques.

Unsurprisingly, the non-Horn reasoning tools, i.e., clingo and SOLAR, are about two orders of magnitude slower than the fastest approach on the artificial examples. Although the methods are inefficient on average, the results show that the applications are dependable methods by solving between 97 % to 99 % of the samples in time for both evaluation sets. From Fig. 1, we can further conclude that for the simpler examples the computation time for *CF* grows faster than for the ASP solver. This situation, however, changes with more runtime expensive diagnosis problems.

For a more in-depth comparison of the algorithms, we created the scatter plots in Figs. 2 and 3. Each data point symbolizes one sample run. The x and y values characterize the penalized log runtimes of the corresponding algorithm pair. For instance, in Fig. 2a, we compare *ATMS* to $HS\text{-}DAG_{QX}$ on *Artificial Samples I*. Every point above the diagonal represents a PHCAP execution, where *ATMS* was more efficient than $HS\text{-}DAG_{QX}$, while every point below the line indicates the superiority of $HS\text{-}DAG_{QX}$ on a diagnosis problem. The dashed lines mark the penalized runtime, i.e., every execution exceeding the runtime limit is located on the dashed lines. Consider Fig. 3a; although there are executions where $HS\text{-}DAG_{QX}$ is rather inefficient on *Artificial Samples II* in comparison to *ATMS*, the data points accumulate on and below the diagonal close to the origin. This suggests that for the bulk of the samples, *ATMS* takes longer to compute the diagnose; this is also indicated by the median runtime results in Table 3 under $Runtimes_{\theta}$. Investigating the threshold lines, we can determine that only a single execution led to a timeout on $HS\text{-}DAG_{QX}$ but not on *ATMS*. All other timed-out data points are exactly at the intersection of the threshold lines, thus representing several instances causing both approaches to exceed the time limit.

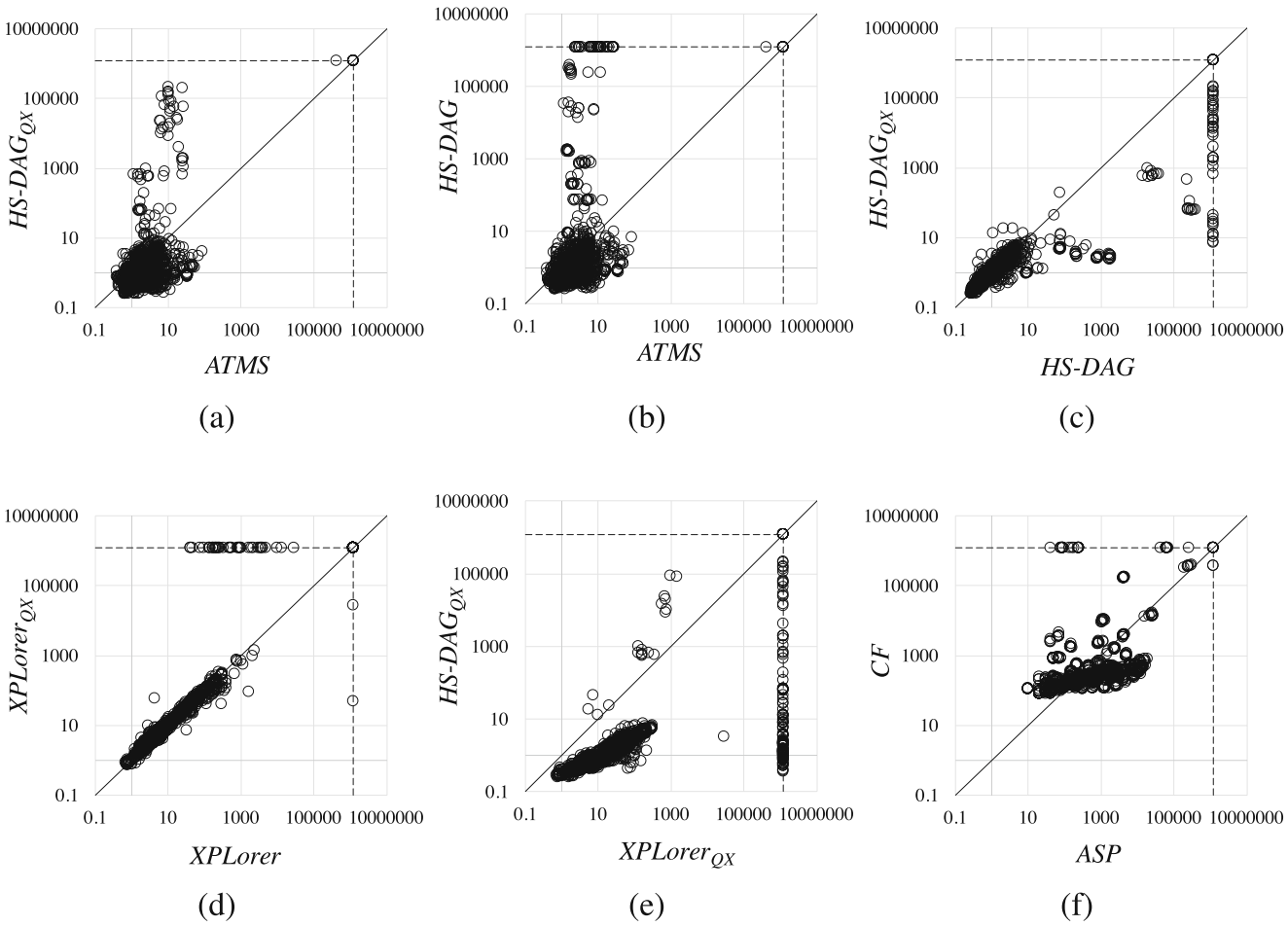


Fig. 3 Scatter plots comparing the penalized runtimes [10^y ms] on *Artificial Samples II*

While *HS-DAG*$_{QX}$ provides more appealing results in comparison to the version without QuickXplain, we cannot observe the same for *XPLorer* and *XPLorer*$_{QX}$. There both conflict extraction methods perform rather similarly. This is apparent from Figs. 2d and 3d, where data points are mostly located symmetrically around the diagonal. Yet the insertion-based extraction of the conflict as suggested by Arif et al.'s [1] solves more samples in time. Specifically, on *Artificial Samples II* only a few samples exist in which *XPLorer* was penalized while *XPLorer*$_{QX}$ provided the solutions in time.

For the two non-Horn solvers, the results on the artificial samples are interesting. On the first set of examples, the consequence finding procedure is preferable, i.e., more samples are solved in time and on median consequence finding is more efficient than using the ASP solver. This is also apparent from Fig. 2f. On *Artificial Samples II*, yet, the situation changes slightly since ASP can solve more samples in time than the *CF* method and provides better average results. Examining the minimum runtimes, we can deduce that the two approaches using off-the-shelfs solvers suffer from a computational overhead. Particularly, there seems to be even more set-up effort required for SOLAR[11] in comparison to clingo.

4.2 Real world samples

Reviewing the computation results for the FMEA-based examples the consequence finding approach is the only method solving all 2,130 executions within the given time frame. This subsequently causes it to provide good results within *Runtimes*$_\theta$ since no additional penalty is added. Figure 4, which displays the cumulative runtimes based on the number of dignosis problem solved without penalty, shows that the method is not competitive in comparison to the faster procedures such as abduction with the *ATMS* or the HS-DAG approaches. On the FMEA samples, the superiority of *ATMS* in comparison to *HS-DAG*$_{QX}$ and the remaining approaches is more noticeable than on the artificial benchmarks.

In addition, we can observe in Fig. 5c and d that the computation times of the variations of HS-DAG and power lattice exploration diverge more in comparison to the artificial samples. Given the runtime plot in Fig. 4 we can determine that the divergence mainly occurs in the last third of samples solved.

Regarding this portion of the graph, we can further discover that *ASP*'s and *CF*'s execution times do not grow as steeply at the end as for the other approaches. Further, in Fig. 5f the bulk of data points is located above

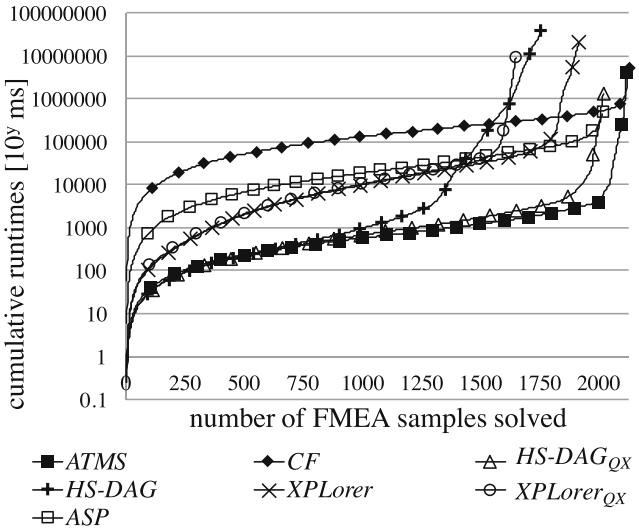


Fig. 4 Numbers of diagnosis samples solved over time for the FMEA diagnosis problems

the diagonal, indicating more samples in which *ASP* was superior. However, several examples could not be solved by *ASP* in time, hence, the less convincing results in *Runtimes*$_\theta$.

4.3 Discussion

Considering the experiments, two approaches seem superior: *ATMS* and *HS-DAG*$_{QX}$. While *ATMS* shows promising results in our set-up, it has to warrant with each added clause all necessary vertex labels are updated, thus, labels have to be checked for consistency[9] and minimality. This is a time-consuming operation especially given a greater number of hypotheses and overlap, which both may lead to larger labels. Hence, for benchmarks with a vast number of assumptions we assume less convincing results from *ATMS*. Comparing the FMEA-based to the artificial samples, the ATMS has to perform more propagations within the latter due the fact that the there might be several levels within the graph, while for the FMEA examples due to their structure it is ensured that there are at most three levels with the last level only containing the explanation node *ex*. In addition, the environments themselves consist of a single element (with the exception of the node *ex*), due to the bijunctive Horn clauses comprising the theory. Yet the labels themselves can become quite large depending on the interconnectedness of the And-or-graph, i.e., manifestations with a large number of causes feature a large label, which has to be considered during minimization and consistency checks. Approaches to focus the ATMS and subsequently avoid label explosion have been proposed [16].

[11] We extract the execution times directly from the solvers' output themselves.

[9] Though in our case no conflicting hypotheses were generated within the PHCAPs.

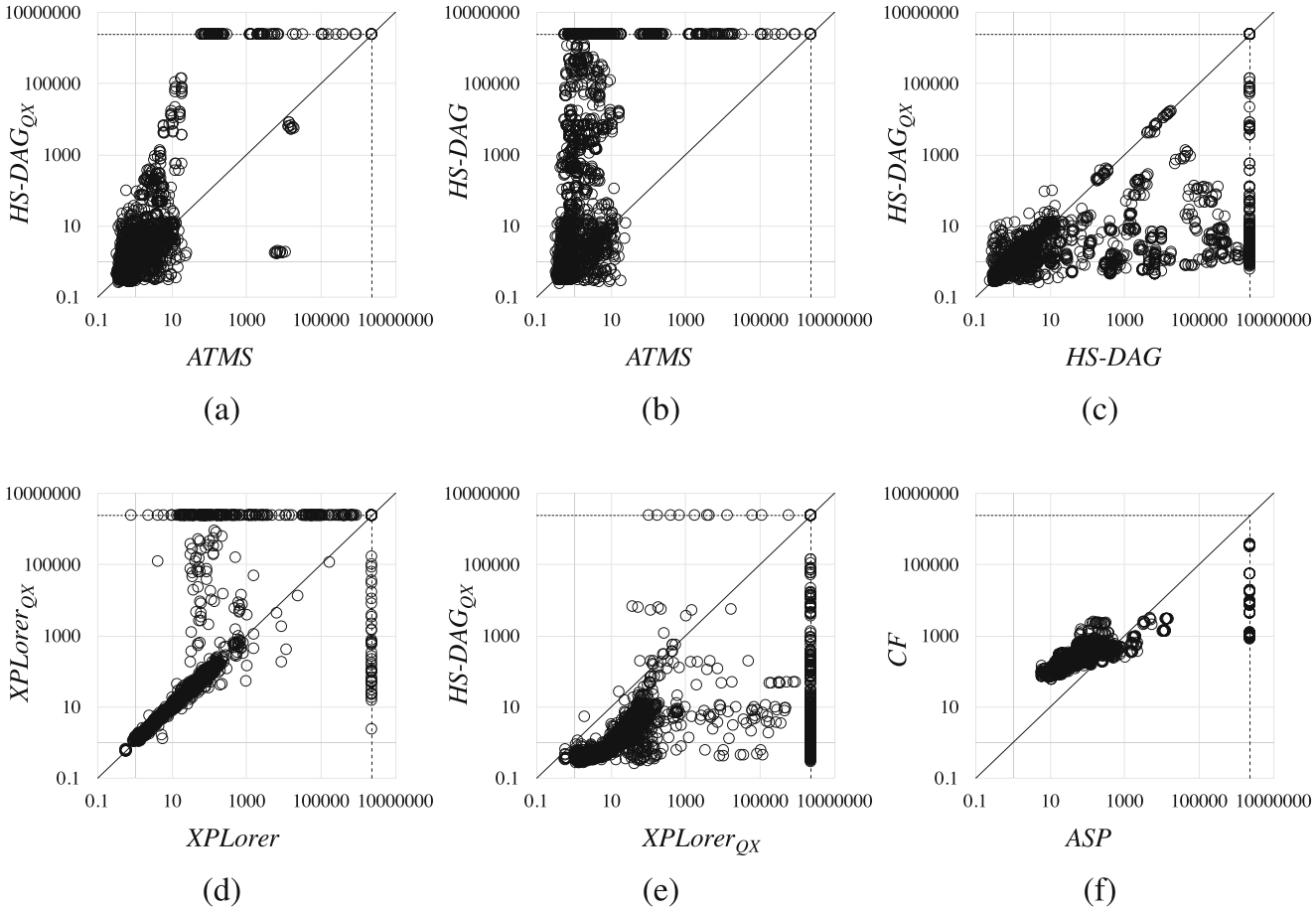


Fig. 5 Scatter plots comparing the penalized runtimes [10^y ms] on *FMEA Samples*

As a side note we would like to mention that the ATMS can also be exploited within a conflict-directed set-up. In fact in the consistency-based variation of MBD, the ATMS records the inconsistencies arising from the health assumptions and observations. Thus, we could use a proof-tree completion approach to derive contradictions recorded within the NOGOOD node of the ATMS. However, to retain consistency, the ATMS has to warrant that each node's label does not contain the conflicts stored in NOGOOD or their supersets. Thus, more consistency checks are necessary in a proof-tree-style abduction with the ATMS than in our current experimental set-up where the NOGOOD node has a small or empty label.

While showing weak runtime results on the artificial samples, SOL-resolution with SOLAR provides solutions for all PHCAPs within *FMEA Samples*. Given the simple structure of the FMEA examples the number of inference steps in the SOL-resolution are diminished in comparison to the artificial models. In particular, for one tableau the number of *resolve*, *skip*, *factor* and *truncation* steps is proportional to the number of observations, while for the artificial samples more reasoning steps may be necessary. Of course the number of tableaux depends on the number of hypotheses inferring the observations. As already mentioned, SOLAR requires some additional set-up time, which is unsurprising since it is not a specialized Horn abduction solver, but designed for first-order clausal theories. This observation relates to clingo, which is a powerful tool suitable for normal as well as disjunctive logic programs. Therefore, within our evaluation set-up abduction with the ASP solver is not competitive. From our analysis we know that the preparation (grounding) and preprocessing time (program simplification) are not the driving factors in the computational effort, but the true solving time is problematic from an efficiency point of view. We assume that we can observe particularly bad results in *Artificial Samples I* due the extensive number of variables, since the saturation technique is based on all variables within the PHCAP.

HS-DAGXPLAIN as well as XPLORER, both restrict the search space by ensuring that known conflicts and their supersets are not considered again during computation. In case of the former, the construction of a pruned DAG already hinders redundant refutations, while blocking

clauses are added to the encoding of the infeasibility map in the latter. Another optimization of XPLORER is to require the seed of the lattice traversal to be a maximal model. This strategy favors contradictions and ensures that the entry point of the conflict search is as "high" as possible in the power set. In contrast in the HS-DAG approach, LTUR generates refutations, which are located somewhere between this maximum sized conflict, as enforced by XPLORER, and a minimal one. Hence, the conflict-driven search via HS-DAG requires less minimization effort than the power set exploration since the starting point of the traversal is "lower". Besides the shrinking of a contradicting set of assumptions to an MUS, which is affected by the size of the diagnosis set, XPLORER spends considerable time to determine the maximal model for the seed. In general, all these factor contribute to the inefficiency of XPLORER within our experiments. A remedy for this might be to use an indirect MUS approach, which first computes the set of MCSes and afterwards derives their minimal hittings sets. These methods have been shown to be efficient for the complete enumeration of refutations [33].

From the results, we conclude that our HS-DAGXPLAIN approach is performing well on the samples, in particular the version which minimizes conflicts right away. Generally, HS-DAGXPLAIN's performance is affected by the size of Δ-Set , i.e., the number of refutations, and the size of the conflicts. Reason being that these two features determine the depth and breadth of the DAG; that is, the larger the conflicts, the more outgoing edges the conflict node has and hence the more child nodes have to be checked for consistency to determine whether they are minimal hitting sets or not. Considering Fig. 3c, *HS-DAG*$_{QX}$ yields more convincing runtime results than *HS-DAG*. By shrinking each contradiction before continuing with the execution on the HS-DAG, the depth of the graph can be reduced as already computed conflicts and their supersets are excluded from further considerations due to the search pattern encoded in the DAG. Thus, the DAG constructed via *HS-DAG* can be larger with more conflicts than the one generated with *HS-DAG*$_{QX}$. Further, *HS-DAG*$_{QX}$ is advantageous in the sense that it can provide results even though the execution has not been concluded, since all conflicts returned already constitute minimal abductive diagnoses given they are consistent. However, as QuickXplain still requires calls to a theorem prover, in our case the assumption-based LTUR, the overall number of consistency checks increases in comparison to *HS-DAG*. Particularly in the case that theorem prover returns an already minimal conflict invoking QuickXplain causes unnecessary consistency checks. Utilizing an approach which already derives minimal conflicts makes the minimization step obsolete. For instance, instead of using LTUR as the theorem prover to infer refutations, we could exploit QuickXplain right away, which returns one minimal conflict [14][12]. Adaptation and improvements of QuickXplain have been proposed such as MergeXplain [48], which returns a set of minimal conflicts. Other possible improvements encompass to adapt HS-DAGXPLAIN to a parallelized version, which significantly improvements performance [23].

Given the structural characteristics of the FMEA-based models we can easily represent the theory as a DAG with a forward structure from causes to effects. This structure is in fact equivalent to the problems in the simple parsimonious set covering theory proposed by Peng and Reggia [43]. In this framework hypotheses and manifestations are disjoint sets and diagnoses are characterized by the set of hypotheses covering the observations. As it has been shown previously, set covering is equivalent to the hitting set problem [26], thus these types of diagnosis problems can be solved quite efficiently by employing HS algorithms [29].

5 Conclusion

Abductive reasoning is an NP-complete problem. Hence, determining algorithms that can compute explanations in an acceptable time frame is an interesting research problem with relevance not only for the diagnosis community, but also all other application areas of abduction. We tackle this problem in this paper by reviewing two problem formulations within the context of propositional Horn clause abduction, i.e., direct proof and conflict-driven methods. We have exploited well-known as well as recent abductive reasoning algorithms and have shown possible adaptations of the techniques to enable a more efficient computation given our context. Besides implementing abductive reasoning procedures we have also included preexisting tools to give a sense of their capability in regard to this specific framework. To reveal performance trends, we created an evaluation set-up based on artificially samples similar to fault knowledge in practice and real world examples stemming from failure assessments.

Our results show that neither the direct nor the conflict-driven approach provides a universal advantage for Horn clause abduction. The two most promising techniques based on our created problem instances were *ATMS* and abduction via HS-DAG with an immediate minimization of conflicts via QuickXplain. These two methods are particularly suitable to compare direct and conflict-driven techniques, since they exploit the same theorem prover. Both could solve a reasonable amount of samples within the given time frame and our data shows that *ATMS* on average is the most efficient approach, while *HS-DAG*$_{QX}$ computes diagnoses

[12] In this case, we could still use LTUR or a SAT solver as QuickXplain's consistency check mechanism.

faster for more instances. Even though the general solvers were around two orders of magnitude slower than the best Horn reasoner, they have shown a consistent performance being able to compute explanations for most samples within the given runtime allowance. These results demonstrate that off-the-shelf tools can very well function as suitable abduction engines despite not being competitive in regard to runtime.

Inspired by infeasibility analysis, we implemented an exploration of the power lattice. While this strategy has accomplished good results in the environment of group-MUS for Horn formulae, we could not confirm this for our problem domain. Yet MUS extraction based on an incremental LTUR can produce slightly better runtime data than QuickXplain. In light of the results, we plan further investigating strategies from infeasibility analysis to improve upon the computation of abductive diagnoses in the context of Horn formulae. In particular, considering MUS enumeration methods in conjunction with HS-DAG might allow to compute diagnoses very efficiently. In addition, considering conflict extraction approaches, such as MergeXplain, that not only derive a single refutation but compute several at once is a possible area for future research.

Funding Information Open access funding provided by Graz University of Technology.

References

1. Arif MF, Mencía C, Marques-silva J (2015) Efficient MUS enumeration of Horn formulae with applications to axiom pinpointing. In: International conference on theory and applications of satisfiability testing. Springer, pp 324–342
2. Bylander T, Allemang D, Tanner MC, Josephson JR (1991) The computational complexity of abduction. Artif Intell 49(1-3):25–60
3. Catelani M, Ciani L, Luongo V (2010) The FMEDA approach to improve the safety assessment according to the IEC61508. Microelectron Reliab 50(9):1230–1235
4. Charniak E, Goldman RP (1991) Probabilistic abduction for plan recognition. Brown University, Department of Computer Science
5. Charwat G, Wallner JP, Woltran S (2013) Utilizing ASP, for generating and visualizing argumentation frameworks. arXiv:1301.1388
6. Console L, Torasso P (1991) A spectrum of logical definitions of model-based diagnosis. Comput Intell 7(3):133–141
7. De Kleer J (1986) An assumption-based TMS. Artif Intell 28(2):127–162
8. Eén N., Sörensson N (2003) An extensible SAT-solver. In: International conferemce on theory and applications of SAT. Springer, pp 502–518
9. Egly U, Eiter T, Tompits H, Woltran S (2000) Solving advanced reasoning tasks using quantified boolean formulas. In: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence. AAAI Press, pp 417–422
10. Eiter T, Gottlob G (1995) The complexity of logic-based abduction. J ACM (JACM) 42(1):3–42
11. Eiter T, Gottlob G (1995) On the computational cost of disjunctive logic programming: Propositional case. Ann Math Artif Intell 15(3):289–323
12. Eiter T, Leone N, Mateis C, Pfeifer G, Scarcello F (1998) The KR system dlv: Progress report, comparisons and benchmarks. In: Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning. Morgan Kaufmann Publishers Inc., pp 406–417
13. Feldman A, Kurtoglu T, Narasimhan S, Poll S, Garcia D (2010) Empirical evaluation of diagnostic algorithm performance using a generic framework. Int J Prognost Health Manag 1:24
14. Felfernig A, Schubert M, Reiterer S (2013) Personalized diagnosis for over-constrained problems. In: Proceedings of the Twenty-Third international joint conference on Artificial Intelligence. AAAI Press, pp 1990–1996
15. Finin T, Morris G (1989) Abductive reasoning in multiple fault diagnosis. Artif Intell Rev 3(2):129–158
16. Forbus KD, Kleer JD (1988) Focusing the ATMS. In: Proceedings of the Seventh AAAI Conference on Artificial Intelligence. AAAI Press, pp 193–198
17. Friedrich G, Gottlob G, Nejdl W (1990) Hypothesis classification, abductive diagnosis and therapy. Expert Systems in Engineering Principles and Applications, pp 69–78
18. Gebser M, Kaminski R, Kaufmann B, Schaub T (2014) Clingo = ASP + control Preliminary report. In: Leuschel M, schrijvers T (eds) Technical communications of the 30th international conference on logic programming, pp 1–13
19. Gordon AS (2016) Commonsense interpretation of triangle behavior. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. AAAI Press, pp 3719–3725
20. Hawkins PG, Woollons DJ (1998) Failure modes and effects analysis of complex engineering systems using functional models. Artif Intell Eng 12(4):375–397
21. Ignatiev A, Morgado A, Marques-Silva J (2016) Propositional abduction with implicit hitting sets. arXiv:1604.08229
22. Inoue K (1992) Linear resolution for consequence finding. Artif Intell 56(2):301–353
23. Jannach D, Schmitz T, Shchekotykhin K (2015) Parallelized hitting set computation for model-based diagnosis. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence. AAAI Press, pp 1503–1510
24. Kakas AC, Kowalski RA, Toni F (1992) Abductive logic programming. J Log Comput 2(6):719–770
25. Kakas AC, Van Nuffelen B, Denecker M (2001) A-system: Problem solving through abduction. In: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, pp 591–596
26. Karp RM (1972) Reducibility among combinatorial problems. In: Complexity of computer computations. Springer, pp 85–103
27. Koitz R, Wotawa F (2015) Finding explanations: An empirical evaluation of abductive diagnosis algorithms. In: Proceedings of the 2015 International Workshop on Defeasible and Ampliative Reasoning-Volume 1423. CEUR-WS. org, pp 36–42
28. Koitz R, Wotawa F (2015) SAT-based abductive diagnosis. In: 26Th international workshop on principles of diagnosis (DX-2015), pp 1–9

29. Koitz R, Wotawa F (2016) On structural properties to improve FMEA-based abductive diagnosis. In: Workshop on knowledge-based techniques for problem solving and reasoning, pp 1–7
30. Koitz-Hristov R (2018) From Theory to Practice: Abductive Model-based Diagnosis and its Industrial Application. PhD thesis, Graz University of Technology
31. Koitz-Hristov R, Wotawa F (2018) Applying algorithm selection to abductive diagnostic reasoning. Appl Intell 48(11):3976–3994
32. Liffiton MH, Previti A, Malik A, Marques-Silva J (2016) Fast, flexible MUS enumeration. Constraints 21(2):223–250
33. Liffiton MH, Sakallah KA (2008) Algorithms for computing minimal unsatisfiable subsets of constraints. J Autom Reason 40(1):1–33
34. Lucas PJ (1998) Analysis of notions of diagnosis. Artif Intell 105(1):295–343
35. Marquis P (2000) Consequence finding algorithms. In: Handbook of defeasible reasoning and uncertainty management systems. Springer, pp 41–145
36. McIlraith SA (1994) Generating tests using abduction. In: Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning. Morgan Kaufmann Publishers Inc., pp 449–460
37. McIlraith SA (1998) Logic-based abductive inference. Knowledge Systems Laboratory, Technical Report KSL-98-19
38. Nabeshima H, Iwanuma K, Inoue K, Ray O (2010) SOLAR An automated deduction system for consequence finding. AI Commun 23(2-3):183–203
39. Ng HT, Mooney RJ (1992) Abductive plan recognition and diagnosis: A comprehensive empirical evaluation. In: Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning. Morgan Kaufmann Publishers Inc., pp 499–508
40. Nica I, Pill I, Quaritsch T, Wotawa F (2013) The route to success - a performance comparison of diagnosis algorithms. In: International joint conference on artificial intelligence (IJCAI), Bejing, China, pp 1039–1045
41. Nordh G, Zanuttini B (2008) What makes propositional abduction tractable. Artif Intell 172(10):1245–1284
42. Peischl B, Wotawa F (2003) Computing diagnosis efficiently: A fast theorem prover for propositional Horn theories. In: Proceedings of the 14th International Workshop on Principles of Diagnosis, pp 175–180
43. Peng Y, Reggia JA (1990) Abductive inference models for diagnostic problem-solving. Springer, Berlin
44. Poole D, Goebel R, Aleliunas R (1987) Theorist: A logical reasoning system for defaults and diagnosis. In: The knowledge frontier. Springer, pp 331–352
45. Reiter R (1987) A theory of diagnosis from first principles. Artif Intell 32(1):57–95
46. Saikko P, Wallner JP, Järvisalo M (2016) Implicit hitting set algorithms for reasoning beyond NP. In: Proceedings of the Fifteenth International Conference on Principles of Knowledge Representation and Reasoning. AAAI Press, pp 104–113
47. Schüller P (2016) Modeling variations of first-order Horn abduction in answer set programming. Fundamenta Informaticae 149(1-2):159–207
48. Shchekotykhin K, Jannach D, Schmitz T (2015) MERGEXPLAIN: fast computation of multiple conflicts for diagnosis. In: Proceedings of the 24th International Conference on Artificial Intelligence. AAAI Press, pp 3221–3228
49. Simon L, Del Val A (2001) Efficient consequence finding. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence-Volume 1. Morgan Kaufmann Publishers Inc., pp 359–365
50. Wei-Kleiner F, Dragisic Z, Lambrix P (2014) Abduction framework for repairing incomplete EL ontologies: complexity results and algorithms. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence. AAAI Press, pp 1120–1127
51. Wotawa F (2014) Failure mode and effect analysis for abductive diagnosis. In: Proceedings of the International Workshop on Defeasible and Ampliative Reasoning, pp 1–13

Roxane Koitz-Hristov is a post doctoral researcher at the Institute for Software Technology at Graz University of Technology. She obtained her PhD degree in Computer Science from Graz University of Technology in 2018. In her research, she mainly focuses on improvements in regard to abductive model-based diagnosis. In particular she has worked on the Austrian Funding Agency project AMOR dealing with bridging the gap between the theory and application of model-based diagnosis.

Franz Wotawa received a M.Sc. in Computer Science (1994) and a PhD in 1996 both from the Vienna University of Technology. He is currently professor of software engineering at the Graz University of Technology. Since the founding of the Institute for Software Technology in 2003 to the year 2009 Franz Wotawa had been the head of the institute. His research interests include model-based and qualitative reasoning, theorem proving, mobile robots, verification and validation, and software testing and debugging. Beside theoretical foundations he has always been interested in closing the gap between research and practice. For this purposes he founded Softnet Austria in 2006, which is a nonprofit organization carrying out applied research projects together with companies. Starting from October 2017, Franz Wotawa is the head of the Christian Doppler Laboratory for Quality Assurance Methodologies for Cyber-Physical Systems. During his career Franz Wotawa has written more than 330 papers for journals, books, conferences, and workshops. He supervised 84 master and 34 PhD students. For his work on diagnosis he received the Lifetime Achievement Award of the Intl. Diagnosis Community in 2016. Franz Wotawa has been member of a various number of program committees and organized several workshops and special issues of journals. He is a member of the Academia Europaea, the IEEE Computer Society, ACM, the Austrian Computer Society (OCG), and the Austrian Society for Artificial Intelligence and a Senior Member of the AAAI.