ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230490

# How Should I Compute My Candidates? A Taxonomy and Classification of Diagnosis Computation Algorithms

**Patrick Rodler**[a]

[a]University of Klagenfurt, Austria

**Abstract.** Model-based diagnosis is a powerful, versatile and well-founded approach to troubleshooting a wealth of different types of systems. Diagnosis algorithms are both numerous and highly heterogeneous. In this work, we propose a taxonomy that allows their standardized assessment, classification and comparison. The aim is to *(i)* give researchers and practitioners an impression of the diverse landscape of available techniques, *(ii)* allow them to easily retrieve and compare the main features as well as pros and cons, and *(iii)* facilitate the selection of the "right" algorithm to adopt for a particular problem case, e.g., in practical diagnostic settings, for comparison in experimental evaluations, or for reuse, modification, extension, or improvement in the course of research. Finally, we demonstrate the value and application of the taxonomy by assessing and categorizing a range of more than 30 important diagnostic methods, and we point out how using the taxonomy as a common guideline for algorithm analysis would benefit the research community in various regards.

## 1 Introduction

Model-based diagnosis is a well-founded, principled approach to detecting, finding, and fixing faults in numerous types of systems. One of the key tasks to this end is the computation of diagnoses (fault hypotheses), which are essential for both fault localization and repair. Due to its generality, the model-based diagnosis formalism has been used to express and tackle a wide diversity of debugging problems in application areas ranging from software [30], recommender systems [19], spreadsheets [32], ontologies [28] and knowledge bases [60] to hardware [21], circuits [13], robots [69], scheduling problems [55], cars [57], and aircrafts [24]. This has led to a remarkable multitude and heterogeneity of the diagnosis computation methods proposed in the literature, which are often motivated by and tailored for application-specific requirements and problem cases. As a result, it is a hard task for researchers and practitioners to

- get an overview of existing approaches,
- identify the crucial properties of diagnostic techniques,
- assess the methods based on these properties, and
- choose the appropriate approach for a (research- or application-related) diagnostic task at hand.

With this work, we account for this by presenting a taxonomy for diagnosis computation algorithms. Specifically, we introduce and formally define a range of features which are arguably vital for a proper understanding, comparison, selection, and use of diagnostic techniques. We explain the influence of each feature on the selection of an algorithm for a diagnostic task, discuss the potential impact of feature manifestations on the performance of diagnosis algorithms, and examine relationships among the features. To demonstrate the value and application of the proposed taxonomy, we provide a multi-dimensional assessment and categorization of numerous important diagnostic methods in the literature.

## 2 Preliminaries

*Model-based diagnosis* [45] assumes a system (e.g., circuit, software, knowledge base) consisting of a set of *components* $\text{COMPS} = \{c_1, \ldots, c_n\}$ (e.g., gates, lines of code, axioms) which is formally (and correctly[1]) described in some monotonic logical language. Beside relevant general knowledge about the system, this *system description* SD includes a specification of the normal behavior (logical sentence $\text{BEH}(c_i)$) of all components $c_i$ of the form $\text{OK}(c_i) \rightarrow \text{BEH}(c_i)$. As a result, when assuming all components to be fault-free, i.e., $\text{OK}(\text{COMPS}) := \{\text{OK}(c_1), \ldots, \text{OK}(c_n)\}$, conclusions about the normal system behavior can be drawn by means of theorem provers. When the real system behavior, ascertained through *system observations* and/or *system measurements* (stated as logical sentences OBS and MEAS), is inconsistent with the behavior predicted by SD, the normality assumption for some component(s) has to be retracted. We call $\langle \text{SD}, \text{COMPS}, \text{OBS}, \text{MEAS} \rangle$ a *diagnosis problem instance (DPI)*.

**Example:** Consider the problem of diagnosing a propositional knowledge base consisting of five axioms $\{ax_1 : A \rightarrow \neg B, ax_2 : A \rightarrow B, ax_3 : A \rightarrow \neg C, ax_4 : B \rightarrow C, ax_5 : A \rightarrow B \vee C\}$, given a single observation, $A$, and no measurements to start with. Then, a corresponding DPI would be defined as follows: $\text{COMPS} := \{c_1, \ldots, c_5\}$, $\text{SD} := \{\text{OK}(c_i) \rightarrow \text{BEH}(c_i) \mid c_i \in \text{COMPS}\}$ where $\text{BEH}(c_i) := ax_i$ for $i \in \{1, \ldots, 5\}$, $\text{OBS} := \{A\}$, $\text{MEAS} := \emptyset$. Note that $\text{SD} \cup \text{OBS} \cup \text{MEAS} \cup \text{OK}(\text{COMPS})$ is inconsistent. Hence, at least one component (axiom in the knowledge base) must be faulty. □

For a DPI, one is interested in finding a diagnosis, i.e., a set of components whose abnormality would explain the observed incorrect system behavior. Formally, a set $\mathcal{D} \subseteq \text{COMPS}$ is called a *diagnosis* iff $\text{SD} \cup \text{OBS} \cup \text{MEAS} \cup \text{OK}(\text{COMPS} \setminus \mathcal{D}) \cup \text{NOK}(\mathcal{D})$ is consistent where $\text{OK}(X) := \{\text{OK}(c_i) \mid c_i \in X\}$ and $\text{NOK}(X) := \{\neg\text{OK}(c_i) \mid c_i \in X\}$. A diagnosis $\mathcal{D}$ is termed *minimal* / *minimum-cardinality* iff there is no diagnosis $\mathcal{D}'$ such that $\mathcal{D}' \subset \mathcal{D}$ / $|\mathcal{D}'| < |\mathcal{D}|$.

**Example (cont'd):** For our example DPI, there are four minimal diagnoses, given by $\mathcal{D}_1 : [c_1, c_3]$, $\mathcal{D}_2 : [c_1, c_4]$, $\mathcal{D}_3 : [c_2, c_3]$,

[1] Techniques for handling violations of this assumption are discussed in [3].

and $\mathcal{D}_4 : [c_2, c_5]$. For instance, $\mathcal{D}_1$ is a diagnosis as the knowledge base $\text{SD} \cup \text{OBS} \cup \text{MEAS} \cup \text{OK}(\text{COMPS} \setminus \mathcal{D}_1) \cup \text{NOK}(\mathcal{D}_1) = \{A, ax_2, ax_4, ax_5\} = \{A, A \rightarrow B, B \rightarrow C, A \rightarrow B \vee C\}$ is consistent, and $\mathcal{D}_1$ is minimal because the addition of $ax_1$ or $ax_3$ (or both) to this knowledge base would imply its inconsistency. Since there is no diagnosis of cardinality one, all diagnoses $\mathcal{D}_1, \ldots, \mathcal{D}_4$ are also minimum-cardinality diagnoses. □

For efficiency and tractability reasons, the focus in model-based diagnosis is often laid on minimal diagnoses only [37]. In particular, the minimal diagnoses are representative of all diagnoses under the *weak fault model* [11], where the system description SD contains only information about the normal behavior of the system components (and leaves the components' behavior undefined in case of failure, as opposed to techniques using strong fault models, e.g., [14, 42]). We restrict the study in this paper to diagnosis computation methods relying on a weak fault model. They address the following problem:

**Problem 1** (Diagnosis Computation)**.**
***Given:*** *A DPI* $\langle \text{SD}, \text{COMPS}, \text{OBS}, \text{MEAS} \rangle$*, an integer* $k \geq 1$*.* ***Find:*** *Find* $k$ *minimal diagnoses (satisfying a property* $p$*) for* $\langle \text{SD}, \text{COMPS}, \text{OBS}, \text{MEAS} \rangle$*.*

Diagnostic techniques may solve different manifestations of this problem. E.g., they might aim at computing $m$ / all minimal diagnoses (by specifying $k := m$ / $k := \infty$), or at finding all minimum-cardinality diagnoses (by specifying $k := \infty$ and $p :=$ minimum-cardinality).

Useful for diagnosis computation and technically closely related to the concept of a diagnosis is the notion of a conflict, which is a set of components such that the assumption of all of them being fault-free is inconsistent with the current knowledge about the system. Formally, a set of components $\mathcal{C} \subseteq \text{COMPS}$ is a *conflict* iff $\text{SD} \cup \text{OBS} \cup \text{MEAS} \cup \text{OK}(\mathcal{C})$ is inconsistent. Again, we call a conflict $\mathcal{C}$ *minimal* iff there is no conflict $\mathcal{C}'$ with $\mathcal{C}' \subset \mathcal{C}$.

**Example (cont'd):** For our example DPI, we have four minimal conflicts, namely $\mathcal{C}_1 : \langle c_1, c_2 \rangle$, $\mathcal{C}_2 : \langle c_2, c_3, c_4 \rangle$, $\mathcal{C}_3 : \langle c_1, c_3, c_5 \rangle$, and $\mathcal{C}_4 : \langle c_3, c_4, c_5 \rangle$. For example, $\mathcal{C}_1$ is a conflict since the knowledge base $\text{SD} \cup \text{OBS} \cup \text{MEAS} \cup \text{OK}(\mathcal{C}_1) = \{A, ax_1, ax_2\} = \{A, A \rightarrow \neg B, A \rightarrow B\}$ is inconsistent, and $\mathcal{C}_1$ is minimal since the elimination of $ax_1$ or $ax_2$ (or both) would imply its consistency. □

Two important links between diagnoses and conflicts are [45]:

*Hitting-Set Property* A (minimal) diagnosis is a (minimal) hitting set of all minimal conflicts.
($X$ is a *hitting set* of a collection of sets $\mathbf{S}$ iff $X \subseteq \bigcup_{S \in \mathbf{S}} S$ and $X \cap S \neq \emptyset$ for all $S \in \mathbf{S}$.)

*Duality Property* $X$ is diagnosis iff $\text{COMPS} \setminus X$ is no conflict.

**Example (cont'd):** Considering the minimal conflicts $\mathcal{C}_1, \ldots, \mathcal{C}_4$ and the minimal diagnoses $\mathcal{D}_1, \ldots, \mathcal{D}_4$ for our example DPI, we can easily verify the Hitting-Set Property by checking that each $\mathcal{D}_i$ is indeed a hitting set of all $\mathcal{C}_j$. Verifying the Duality Property for, e.g., $\mathcal{D}_4$, means to check if $\text{COMPS} \setminus \mathcal{D}_4$ is not a conflict. This can be seen by observing that $\text{SD} \cup \text{OBS} \cup \text{MEAS} \cup \text{OK}(\text{COMPS} \setminus \mathcal{D}_4) = \{A, ax_1, ax_3, ax_4\} = \{A, A \rightarrow \neg B, A \rightarrow \neg C, B \rightarrow C\}$ is consistent. □

In many cases, there is a substantial number of competing diagnoses. The goal is then to isolate the *actual diagnosis* which pinpoints the *actually* faulty components. Basically, two strategies exist to handle multiple diagnoses, aiming at reducing or avoiding the effort for a manual inspection of the diagnoses: *(i) rank or restrict the computed diagnoses* based on some informative criterion such as maximal probability or minimal cardinality [37], or *(ii) apply sequential diagnosis techniques* to acquire additional information about the diagnosed system to gradually refine the set of diagnoses [13, 53]. Whereas rankings or focusing techniques can be very powerful if the actual diagnosis appears (early) in the solution list, there is no guarantee that the actually faulty components will be located (efficiently).

The more sophisticated sequential diagnosis techniques, on the other hand, gather further system measurements (MEAS) to iteratively rule out spurious diagnoses. They aim at solving the following problem (with highest efficiency):

**Problem 2** (Sequential Diagnosis)**.**
***Given:*** *A DPI* $\langle \text{SD}, \text{COMPS}, \text{OBS}, \text{MEAS} \rangle$*.* ***Find:*** *A sequence of measurements (i.e., logical sentences)* $m_1, \ldots, m_k$*, such that there is a single (highly probable) minimal diagnosis for* $\langle \text{SD}, \text{COMPS}, \text{OBS}, \text{MEAS} \cup \{m_1, \ldots, m_k\} \rangle$*.*

Many sequential diagnosis methods can be characterized by a recurring execution of four phases [12]: *(1)* computation of a set of (preferred, e.g., most probable) minimal diagnoses, *(2)* selection of a (most informative) system aspect to be measured based on the given diagnoses, *(3)* conduction of measurement actions (by some oracle, e.g., an engineer if a circuit is diagnosed), and *(4)* exploitation of the measurement outcome to formulate a new measurement (logical sentence) $m_i$ to update the system knowledge. That is, the DPI is modified (by extending MEAS) between each two iterations of these phases. The execution stops if Problem 2 is solved, i.e., sufficient diagnostic certainty is achieved.

**Example (cont'd):** Assume we would like to impose a ranking on the minimal diagnoses $\mathcal{D}_1, \ldots, \mathcal{D}_4$ obtained for our example DPI. Then, in this particular case, a ranking criterion based on cardinality would not be helpful as all diagnoses have the same cardinality two. We could however draw on a notion of axiom fault probability determined by means of the syntactic complexity of axioms (cf., e.g., [59, 46]). For instance, we might know from experience that knowledge engineers are usually much more prone to faults when using logical negation rather than disjunction or implication. Based on this assumption, diagnoses would be the more likely to pinpoint the actual fault in the knowledge base the more axioms with negation they involve. A possible ranking (from more to less likely) obtained from this analysis would be $[\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4]$.

Alternatively, one might choose to provably rule out spurious diagnoses by adding new measurements to the DPI. Employing sequential diagnosis techniques to our example DPI could yield the query to a knowledgeable user (e.g., a domain expert) whether $B$ or $\neg B$ must hold. Assuming the answer $B$ and the addition of $B$ to MEAS, we would find that all minimal conflicts for the new DPI are given by $\mathcal{C}_5 := \langle c_1 \rangle$ and $\mathcal{C}_6 := \langle c_3, c_4 \rangle$, which would disprove $\mathcal{D}_3$ and $\mathcal{D}_4$ and give rise to only two remaining minimal diagnoses, $\mathcal{D}_1$ and $\mathcal{D}_2$ (all minimal hitting sets of $\mathcal{C}_5, \mathcal{C}_6$). In a final step, to distinguish between $\mathcal{D}_1$ and $\mathcal{D}_2$, a sequential diagnosis method could ask the user about the truth value of $C$, where a positive (negative) answer would prove that $\mathcal{D}_1$ ($\mathcal{D}_2$) is the correct diagnosis, i.e., that the axioms $ax_1$ and $ax_3$ ($ax_1$ and $ax_4$) are actually faulty. □

## 3 A Taxonomy for Diagnosis Algorithms

We now propose a collection of pivotal features of diagnosis computation algorithms, based on which we will classify and compare important existing techniques in Sec. 4 and Tab. 1. In the following, we assume that an algorithm $A$ addresses (some manifestation of) the diagnosis computation problem (cf. Problem 1) and is given as input a DPI and possibly some meta information (such as component failure rates that allow to derive diagnosis probabilities [13], or

algorithm-specific parameters, e.g., stop, pruning or restart criteria [16, 1]).[2] We describe each feature by giving a *definition* of its possible manifestations, a brief explanation of its *relevance* wrt. algorithm selection for a diagnostic task, a short discussion of the practical *impact* of different feature manifestations, and a comment on the *relationship* to other features. The features can be logically grouped into five categories, i.e., *Output Qualities* (Bullets **1**–**4** below), *Way of Computation* (**5**–**6**), *Sequential Diagnosis Context* (**7**–**8**), *Application Context* (**9**–**11**), and *Performance* (**12**), as shown in Tab. 1 and discussed next.

### *Output Qualities*

**(1) Soundness:** *Definition:* $A$ is *sound* iff it outputs only minimal diagnoses; otherwise, it is *unsound*. *Relevance:* Soundness is necessary if *(a)* no actually healthy components should be marked as faulty in a diagnostic scenario, e.g., when inspecting or changing parts unnecessarily is costly such as in a car [27], or when a modification of correct components impacts the quality of the system such as for axioms in a knowledge base [47], or if *(b)* every solution returned by the algorithm should indeed be a possible explanation for the observed system misbehavior, e.g., to avoid the necessity of additional computations and a potentially costly post-processing of the solutions. Apart from that, the soundness requirement is in line with the generally accepted principles of parsimony [45] and Occam's razor [4], which postulate that from two different (fault) explanations, the simpler one is preferable. *Impact:* Forgoing the requirement of soundness can lead to a higher efficiency of diagnosis computation, as certain unsound algorithms are designed to drop soundness to the benefit of performance (e.g., [38, 16]). There are basically two forms of unsoundness for returned diagnoses, i.e., they might be *(a)* non-minimal diagnoses (intuitively: "too large" component sets; cf., e.g., [16]), or *(b)* non-diagnoses (intuitively: "too small" component sets; cf., e.g., [38]). Both cases can be handled by a suitable post-processing of the returned solutions (cf., e.g., [34]), the cost of which depends on the number of solutions that are non-(minimal) diagnoses and on their degree of unsoundness (i.e., how much "too small" or "too large" they are). *Relationship:* Unsoundness can entail incompleteness (cf. Bullet **2**) or a violation of the best-first property (cf. Bullet **3**), e.g., for systematic hitting set searches [47].

**(2) Completeness:** *Definition:* If $A$ computes a set of diagnoses, it is *all-complete* iff it outputs all minimal diagnoses given sufficient time and memory, and it is *property-complete* iff it outputs all minimal diagnoses with a particular property (e.g., minimum cardinality) given sufficient time and memory. If $A$ computes a single diagnosis (with a particular property $p$, e.g., minimum-cardinality), it is *one-complete* iff it outputs a minimal diagnosis (with property $p$) whenever such a diagnosis exists. Else, if there is any minimal diagnosis that $A$ might fail to compute, it is *incomplete*. *Relevance:* Completeness is necessary if it is crucial in a diagnostic scenario that the actual diagnosis is found with certainty, or when missing the actual diagnosis or a diagnosis with a particular property might have serious consequences, e.g., when diagnosing critical systems such as aircrafts, medical ontologies or security software. Moreover, completeness is vital for reasonable stop conditions in sequential diagnosis scenarios, e.g., if a single diagnosis remains after taking some measurements, only completeness implies that this diagnosis is the only possible minimal fault. *Impact:* Forgoing the requirement of completeness allows for a higher efficiency of diagnosis computation in many cases, as incomplete algorithms are often devised with a particular focus on performance, cf., e.g., [16, 1, 38]. *Relationship:* If not carefully devised, incomplete methods are usually not best-first (cf. Bullet **3**).

**(3) Best-First Property:** *Definition:* $A$ is *generally best-first* iff it computes and outputs diagnoses in order according to a given sorting criterion (often: minimal cardinality or maximal probability); $A$ is *focused best-first* iff it is best-first only for a particular sorting criterion (often: minimal cardinality); $A$ is *only-best* iff it computes only the best diagnosis (if its type is single-solution, cf. Bullet **4**) / diagnoses (if its type is multiple-solution, cf. Bullet **4**) wrt. a particular property (often: minimum-cardinality); $A$ is *best-subset-no-order* iff it computes a subset of all diagnoses including exactly the best diagnoses wrt. a particular property (often: minimal cardinality), but the diagnoses are not computed or output in best-first order; $A$ is *any-first* iff it does not satisfy any of the above conditions and cannot guarantee any particular output order of diagnoses; if $A$ is any-first, but uses heuristic techniques to approximate a particular order of the computed diagnoses, it is *heuristic best-first*. *Relevance:* The best-first property is useful, e.g., if (one expects) there is a large number of diagnoses and the actual diagnosis is likely among the best diagnoses (e.g., when all components fail with an equal small likelihood [37]), if focusing techniques are adopted where only the best subset of all diagnoses is further considered [37], if informative samples for sequential diagnosis should be computed [51], or if users intend to monitor the best diagnoses throughout the debugging process [50]. *Impact:* Forgoing the best-first requirement usually leads to a higher computation efficiency, as any-first algorithms can use more performant (e.g., depth-first [60] instead of breadth-first [45] or uniform-cost [47]) diagnosis search techniques. Also, generally best-first methods might be significantly more expensive than related focused best-first ones (cf., e.g., [50]). *Relationship:* To the best of our knowledge, all generally best-first algorithms are conflict-dependent (cf. Bullet **5**), i.e., rely on a systematic search based on conflicts (cf. Tab. 1). Moreover, compilation-based approaches (cf. Bullet **5**) are usually only-best techniques wrt. minimum-cardinality diagnoses (cf. Tab. 1).

**(4) Type of Output:** *Definition:* $A$ is called *multiple-solution* iff it can compute a set of two or more diagnoses per call; otherwise, if $A$ returns at most one diagnosis, it is called *single-solution*. *Relevance:* The optimal algorithm choice wrt. this feature is trivial and depends on whether one or multiple diagnoses are required in a scenario. Note, most algorithms considered in Tab. 1 can output multiple solutions. Clearly, any such algorithm can also be employed if only a single solution is desired. *Impact:* Single-solution techniques can be highly performant as they may use optimizations that harm completeness by manipulating the set of all solutions for the benefit of computational efficiency [10]. To allow for some degree of control over their performance, multiple-solution approaches are sometimes also configurable, e.g., to compute a number of exactly $k$ solutions, to stop after some timeout occurs, to prune a specified part of the search space, or to stop after a predefined number of search iterations have been performed [50, 49, 54, 16, 1]. *Relationship:* Single-solution methods are usually one-complete (cf. Bullet **2**) and only-best (cf. Bullet **3**), see Tab. 1. Simply put, when focusing on only one solution, approaches normally aim at finding the *best* diagnosis wrt. some property among *all* minimal diagnoses.

[2] Works describing diagnostic methods draw on a variety of notations and formalisms, which can however also be expressed using Reiter's general theory [45], as reviewed in Sec. 2.

*Way of Computation*

**(5) Conflict Dependency:** *Definition:* $A$ is *conflict-dependent* iff it requires the computation of (minimal) conflicts; otherwise, i.e., if $A$ works without taking information about conflicts into account, it is *direct*; a direct algorithm which translates the DPI into a target language and performs diagnosis computation based on this alternative problem representation is called *compilation-based*. *Relevance:* If conflict computation or theorem proving is very expensive in a diagnostic scenario (cf., e.g., [55]), then direct algorithms are preferable, or even the only feasible approach. If a systematic exploration of diagnoses (allowing, e.g., inferences that all diagnoses with a specific property, say minimum-cardinality, are already computed) is desired [37] or the general best-first property (see Bullet **3**) is relevant [50], or the used method should be optimized for certain sequential diagnosis problems [52], then a conflict-dependent approach might be the only viable choice. When adopting a conflict-dependent method, it is important to note that an adequate conflict generation approach—which is sound and complete wrt. the computation of (minimal) conflicts as well as applicable to and performant for the DPI at hand—needs to be combined with the diagnosis algorithm. Choosing such an approach might not be an easy task for non-expert users. *Impact:* Compilation-based techniques often allow to answer important diagnostic queries (such as minimum-cardinality diagnosis computation) in polynomial time in the size of the compilation (which however might be exponential). Hence, these methods might be the best choice given that the DPI at hand is amenable to a compact compiled representation. Direct techniques, some of which (e.g., [20, 60]) are based on the Duality Property (cf. Sec. 2), sometimes allow to escape computational bottlenecks concerning memory consumption [60] or time [16] by forgoing a *systematic* enumeration of the diagnoses. Most of the algorithms in the literature appear to be conflict-dependent (cf. Tab. 1), and most (but not all, e.g., [66]) of them are based on the Hitting-set Property (cf. Sec. 2); hence, there is a great selection of such methods, which cover numerous different combinations of other features, so that there will be a reasonable choice among them for many diagnosis tasks and applications. *Relationship:* Considering the literature, it appears that conflict-dependency implies (if judgeable) the general applicability of an algorithm (cf. Bullet **9** and Tab. 1). The reason for this is that, for diagnosis computation, the set of minimal conflicts is representative of a DPI (cf. [9, Theorem 1]), and thus can be seen as a kind of general abstraction from the DPI, which can be applied to any DPI. Hence, algorithms relying on this abstraction (usually) do not make assumptions about system specifics.

**(6) Way of Conflict Computation:** *Definition: (Prerequisite: $A$ is conflict-dependent, cf. Bullet **5**)* $A$ is *preliminary* iff it requires (a non-empty, non-singleton subset of) all minimal conflicts to be given as an input, or if it computes (a non-empty, non-singleton subset of) all minimal conflicts preliminarily, before the diagnosis computation starts; otherwise, if conflicts are computed on-demand in the course of diagnosis computation, $A$ is *on-the-fly*. *Relevance:* If the prior generation of all minimal conflicts is feasible or even practical in a diagnosis scenario, there are highly efficient preliminary techniques available for diagnosis computation (cf., e.g., [61, 10]). These preliminary techniques can also benefit from insights of a significant body of research regarding the minimal hitting set problem (cf. [22] for an overview). On-the-fly algorithms, on the other hand, are often still efficiently applicable even if preliminary conflict generation is infeasible. That said, it might in certain diagnostic use cases not be necessary to explicitly derive all (minimal) conflicts, e.g., in sequential diagnosis scenarios [37, 48] where only a subset of diagnoses is required per iteration. Some preliminary techniques (e.g., [13]) can be modified to act on-the-fly, but not all of them (e.g., ones that exploit the structure in the collection of minimal conflicts [70]). Any on-the-fly algorithm can be modified to be preliminary in a straightforward way (by pre-computing the collection of minimal conflicts and by choosing appropriate conflicts from this collection on-the-fly). *Impact:* Forgoing the preliminary computation of the (full) set of minimal conflicts and intermixing conflict generation with diagnosis computation can allow to escape a combinatorial explosion and thus enhance the performance of diagnosis methods [37]. *Relationship:* Usually, preliminary algorithms do not incorporate mechanisms for generating minimal conflicts, but assume them to be given a priori (e.g., [39, 70, 10]). For such methods, we cannot assess the features general applicability, black-box reasoning, and logics-agnosticism (cf. Bullets **9**, **10** and **11**) as these methods do not directly use the DPI, but require some "external" technique to provide the required collection of conflicts, where the three said features above depend on the adopted conflict generation technique.

*Sequential Diagnosis Context*

**(7) Focus on Sequential Diagnosis:** *Definition:* $A$ is *sequential* iff it provides mechanisms to address the sequential diagnosis problem (cf. Problem 2), e.g., in terms of measurement proposal techniques or system knowledge update procedures after measurement actions; otherwise, $A$ is *one-shot*. *Relevance:* To solve a sequential diagnosis problem, algorithms devised specifically for this purpose will often be more practical than iteratively re-invoking a one-shot algorithm for the various DPIs (successively extended by new measurements, cf. Sec. 2) during a sequential diagnosis session (cf., e.g., [49, 62]). Apart from that, the former techniques will often be directly applicable to a sequential diagnosis task, whereas a user might need to adapt the implementation of a one-shot algorithm to make it ready for sequential diagnosis. On the other hand, if sequential diagnosis is not the task in a diagnostic scenario, then a user is generally better off (wrt. efficiency, implementation complexity, etc.) when using one of the often less sophisticated (cf., e.g., [52]) one-shot techniques. *Impact:* Relying on sequential techniques will usually boost the performance of diagnosis computation in a sequential setting, but will generally also tend to worsen the performance in non-sequential settings. *Relationship:* Sequential techniques are usually sound, complete and stateful (cf. Bullets **1**, **2** and **8**, and Tab. 1) where the former two properties can be useful for diagnostic decision-making (e.g., measurement proposal, stop criteria) and the latter can improve the time performance of an algorithm (cf. [52, 13]).

**(8) Maintenance of State:** *Definition:* $A$ is *stateful* iff it can maintain its state when used throughout a sequential diagnosis process (e.g., by storing or reusing data structures, intermediate values, etc.); otherwise, $A$ is *stateless*. *Relevance:* Since this feature describes the internal workings of an algorithm, users might basically be indifferent whether the used diagnosis method is stateful or stateless. However, requirements wrt. the algorithm's performance may (ceteris paribus) have a bearing on the proper choice between stateful and stateless algorithms (see below). *Impact:* When memory is the more critical resource, e.g., on small or mobile devices, stateless algorithms may be a way to trade more time for less space, whereas, when time is the more critical resource, stateful algorithms may be preferable [52, 54]. *Relationship:* Stateless algorithms are usually (but not always, cf. [60]) one-shot (cf. Bullet **7**), and algorithms that can be used in a stateful way are normally (but not always, cf. [41]) sequential (cf. Bullet **7**). See Tab. 1.

### *Application Context*

**(9) General Applicability:** *Definition:* $A$ is *generally applicable* iff it can be used for any diagnosis problem expressible by means of Reiter's theory [45], i.e., for any DPI as specified in Sec. 2; otherwise, e.g., if $A$ makes certain assumptions about (e.g., the structure or some properties of) the tackled DPI, it is *problem-dependent*. *Relevance:* The appropriate choice of diagnosis algorithm depends on its application area and scope. E.g., if only certain system types (such as circuits) are addressed, then a problem-dependent algorithm that considers and leverages the peculiarities of this system type will be the proper and often much more performant approach (cf. [18, 62, 41]). If, on the other hand, a diagnosis system's intended use is for frequently changing application domains (e.g., in the Semantic Web context, where a multitude of different domains are modeled in terms of ontologies with highly heterogeneous content, structure, expressiveness, reasoning complexity and used logical languages [59, 47, 50]), problem-dependent techniques might not be eligible and generally applicable ones allow to deal with various diagnosis problems without modifying the diagnosis system. *Impact:* If general applicability is required, the price to pay for this is the use of general-purpose diagnostic techniques which naturally cannot match up in terms of performance to approaches geared to optimizing diagnostic efficiency for specific problem cases. *Relationship:* General applicability implies logics-agnosticism (cf. Bullet **11**) since an algorithm incapable of dealing with some (monotonic) logical language is, by definition, not generally applicable. However, there might be logics-agnostic techniques which exploit structural properties of a particular system type (regardless of how it is modeled), which are thus not generally applicable. Moreover, most (but not all, cf. [13]) of the generally applicable methods are black-box (wrt. reasoning), i.e., can use an arbitrary (sound and complete) inference mechanism (cf. Bullet **10**).

**(10) Black-Box Reasoning:** *Definition:* $A$ is *black-box* (wrt. reasoning) iff it uses a reasoner as a black-box oracle (for consistency or model checking) and can use an arbitrary (sound and complete) reasoner for the logical language used to express the DPI; otherwise, if $A$ requires additional computations or mechanisms (e.g., operations pertinent to a specific problem representation [65, 6] or bookkeeping techniques [13]) from a reasoner beyond the main reasoning result, it is *reasoner-dependent* (sometimes also referred to as *glass-box*, cf., e.g., [43, 28]). *Relevance:* If the logic used to model the diagnosed system is stable in an application area, then reasoner-dependent approaches might be the better choice as they might be advantageous in terms of diagnostic efficiency (given a suitable "glass-box" reasoner for the respective logic). E.g., when DPIs expressible by means of propositional logic are the target use case of a diagnosis system, then a reasoner-dependent algorithm based on propositional logic might be preferable to a black-box one with otherwise equal features. If, on the other hand, different formalisms might be used to describe the faulty system [50], black-box techniques can be more expedient. They can always simply use the best reasoner for the particular problem at hand without needing to incorporate any modifications into the reasoner (or the diagnosis algorithm)—unlike reasoner-dependent approaches, which require the incorporation of the necessary additional mechanisms into any adopted reasoner. And, performances of various reasoners might differ substantially [23]. As a rule of thumb, if the performance for one *fixed* system description language should be maximized, then reasoner-dependent approaches tend to be more favorable, whereas black-box methods tend to be preferable if the performance over *variable* modeling languages should be optimized. *Impact:* Reasoner-dependent techniques can lead to an improved time performance [28, 35], e.g., when reasoners extract conflicts as a byproduct of consistency checks [35], or store sets of logical sentences sufficient for derived entailments to hold [13], but might also incur memory overheads [36]. Advantages of black-box methods are, e.g., their robustness (no sophisticated, and potentially error-prone, modifications of complex reasoning algorithms), their simplicity (internals of reasoner irrelevant), their flexibility (e.g., black-box methods can use a portfolio reasoning approach by switching to the most efficient reasoner in a simple plug-in fashion depending on the language used to describe the diagnosed system [56]), and their up-to-dateness (black-box methods can directly benefit from advances in the general research on automated reasoning). *Relationship:* There are no general implications on other features resulting from the presence or absence of the black-box property; however, it is often (but not always) the case that black-box techniques are also generally applicable, logics-agnostic, sound, complete and multiple-solution (cf. Bullets **1**, **2**, **4**, **9** and **11**).

**(11) Logics-Agnosticism:** *Definition:* $A$ is *logics-agnostic* iff it can deal with DPIs expressed by arbitrary (monotonic) logics; otherwise, $A$ is *logics-dependent*. *Relevance:* If a diagnosis approach is intended to be used with only one *fixed* system description language, then a user should choose the method with (expected/reported) best performance for the faced diagnostic task, regardless of whether it is logics-agnostic or -dependent. Logics-dependent approaches, however, can offer very attractive features, e.g., compilation-based approaches (e.g., [41, 65, 6]) can often compute diagnoses in polynomial time once the DPI has been compiled into a target representation; however, they are restricted to propositional logic system descriptions. If a diagnosis approach needs to deal with *diverse* system modeling languages, the adoption of a logics-agnostic method might be the only choice. *Impact:* As our literature study suggests, logics-dependent approaches are usually particularly attractive wrt. performance. The reason is that these methods often use sophisticated (e.g., representation, optimization, or reasoning) techniques specific to one logic (often: propositional logic, cf. Tab. 1). *Relationship:* The property of logics-agnosticism appears to co-occur with the black-box property (cf. Bullet **10**) in most (but not all, cf. [13]) cases. And, logics-agnostic techniques are often generally applicable (cf. Bullet **9**).

### *Performance*

**(12) Space Efficiency:** *Definition:* $A$ is *space-efficient* iff its worst-case memory complexity is polynomial in the input size; otherwise, $A$ is *space-inefficient*. *Relevance:* If time is the resource to be minimized and the diagnostic task is expected to manage with the available memory, then space-inefficient methods can be the better choice, due to their generally lower time complexity. If, on the other hand, the available memory capacity of an (e.g., mobile) device is low or the diagnostic task is expected to be memory-intensive (e.g., if high-cardinality diagnoses exist [60]), then space-efficient algorithms might be the only viable approach (as memory consumption increases exponentially with the size of diagnoses for many space-inefficient techniques). Note, only few space-efficient diagnosis computation strategies have been proposed in the literature, thus there is not plenty of choice, which is why a trade-off between space-efficiency and other properties might be necessary. Only recently has a space-efficient algorithm exhibiting all desirable properties wrt. the features discussed here, along with a reasonable time performance, been suggested [50]. *Impact:* Space-efficiency is often bought for a

| Technique | | | Features | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Output Qualities | | | | Way of Computation | | Seq. Diag. Context | | Application Context | | | Performance |
| Name | Work | Year | SND | COMPL | BEST-F | MULT | CONF-DEP | O-T-FLY | SEQ | STATE | GEN-APP | BL-BOX-RS | ANY-LOG | POLY-SPACE |
| GDE | [13] | 1987 | ✓ | ✓ (all) | ✓ (gen) | ✓ | ✓ | × | ✓ | ✓ | ✓ | × (bk) | ✓ | × |
| HS-Tree | [45] | 1987 | ✓ | ✓ (all) | ✓ (foc=mc) | ✓ | ✓ | ✓ | × | × | ✓ | ✓ | ✓ | × |
| HS-DAG | [25] | 1989 | ✓ | ✓ (all) | ✓ (gen) | ✓ | ✓ | ✓ | × | × | ✓ | ✓ | ✓ | × |
| DIAGNOSE | [29] | 1994 | ✓ | ✓ (all) | × | ✓ | ✓ | × | ✓ | ✓ | ✓ | ✓ | ✓ | × |
| HST | [67] | 2001 | ✓ | ✓ (all) | ✓ (foc=mc) | ✓ | ✓ | ✓ | × | × | ✓ | ✓ | ✓ | × |
| DNNF | [6] | 2001 | ✓ | ✓ (p=mc) | ✓ (only=mc) | ✓ | × (cp-b=DNNF) | na | × | ? | × | × | × (PL) | × |
| Genetic Alg. | [38] | 2002 | × | × | × | ✓ | ✓ | × | × | × | na | na | na | ? |
| BHS-Tree | [39] | 2003 | ✓ | ✓ (all) | × | ✓ | ✓ | × | × | × | na | na | na | × |
| Bool. Alg. | [39] | 2003 | ✓ | ✓ (all) | × | ✓ | ✓ | × | × | × | na | na | na | × |
| HSSE-Tree | [68] | 2006 | ✓ | ✓ (all) | ✓ (foc=mc) | ✓ | ✓ | × | × | × | na | na | na | × |
| HA* | [18] | 2006 | ✓ | ✓ (one=mc) | ✓ (only=mc) | × | × (cp-b=h-DNF) | na | × | ? | × (circ) | ✓ | × (PL) | × |
| OBDD | [65] | 2006 | ∼ (1) | ✓ (all) | ✓ (only=mc) | ✓ | × (cp-b=OBDD) | na | ✓ | ✓ | × | × | × (PL/HL) | × |
| CDA* | [66] | 2007 | ✓ | × | ✓ (gen) | ✓ | ✓ | ✓ | × | ? | ✓ | ✓ | ✓ | × |
| SAFARI | [16] | 2008 | ∼ (2) | × | × | ✓ | × (dir) | na | × | × | ✓ | ✓ | ✓ | × |
| STACCATO | [1] | 2009 | ? | ∼ (3) | × (heur) | ✓ | ✓ | × | × | × | na | na | na | ✓ |
| NGDE | [9] | 2009 | ✓ | ✓ (p=mc) | ✓ (only=mc) | ✓ | ✓ | ✓ | × | ? | ✓ | × (bk) | ✓ | ? (4) |
| Recurs. MHS | [61] | 2010 | ✓ | ✓ (one=mc) | ✓ (only=mc) | × | ✓ | × | × | na | na | na | na | ∼ (5) |
| SDA | [62] | 2011 | ✓ | ✓ (one=SD-sol) | ✓ (only=SD-sol) | × | × (cp-b= {BN, d-DNNF}) | na | ✓ | ✓ | × (circ) | × | × (PL) | × (6) |
| cminc | [10] | 2011 | ✓ | ✓ (one=mc) | ✓ (only=mc) | × | ✓ | × | × | na | na | na | na | ? (7) |
| FastDiag | [20] | 2011 | ✓ | ✓ (all) | × | ✓ | × (dir) | na | × | × | ✓ | ✓ | ✓ | ✓ |
| SDE | [64] | 2012 | ✓ | ✓ (all) | ✓ (foc=mc) | ✓ | ✓ | ✓ | × | × | ✓ | ✓ | ✓ | × |
| Improved Bool. Alg. (*) | [44] | 2012 | ✓ | ✓ (all) | ✓ (bsno) | ✓ | ✓ | × | × | na | na | na | na | × |
| Inv-HS-Tree | [60] | 2014 | ✓ | ✓ (all) | × | ✓ | × (dir) | na | ✓ | × | ✓ | ✓ | ✓ | ✓ |
| SATbD | [41] | 2014 | ✓ | ✓ (p=mc) | ✓ (only=mc) | ✓ | × (cp-b=SAT) | na | × | ✓ | × (circ) | ✓ | × (PL) | ? |
| Increm-distrib-MHS (*) | [70] | 2015 | ✓ | ✓ (all) | × | ✓ | ✓ | × | ✓ | ✓ | na | na | na | × |
| Unif-cost HS-Tree (*) | [47] | 2015 | ✓ | ✓ (all) | ✓ (gen) | ✓ | ✓ | ✓ | × | × | ✓ | ✓ | ✓ | × |
| Parallel HS-Tree | [33] | 2016 | ✓ | ✓ (all) | ✓ (foc=mc) | ✓ | ✓ | ✓ | × | × | ✓ | ✓ | ✓ | × |
| StaticHS | [54] | 2018 | ✓ | ✓ (all) | ✓ (gen) | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | × |
| MOA | [17] | 2020 | ✓ | ✓ (p=mc) | ✓ (only=mc) | ✓ | × (cp-b=SAT) | na | × | × | × | ✓ | × (PL) | × |
| DynamicHS | [49] | 2020 | ✓ | ✓ (all) | ✓ (gen) | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | × |
| D-CMMO | [71] | 2022 | ✓ | ✓ (one=mc) | ✓ (only=mc) | × | × (cp-b=SAT) | na | × | ✓ | × (circ) | ✓ | × (PL) | × |
| RBF-HS | [50] | 2022 | ✓ | ✓ (all) | ✓ (gen) | ✓ | ✓ | ✓ | × | × | ✓ | ✓ | ✓ | ✓ |
| HBF-HS | [50] | 2022 | ✓ | ✓ (all) | ✓ (gen) | ✓ | ✓ | ✓ | × | × | ✓ | ✓ | ✓ | × |
| Heuristic Inv-HS-Tree (*) | [51] | 2022 | ✓ | ✓ (all) | × (heur) | ✓ | × (dir) | na | × | × | ✓ | ✓ | ✓ | ✓ |

**Table 1**: Classification of some important existing diagnosis computation algorithms based on their characteristics wrt. the features proposed in Sec. 3. Table rows are sorted by the year of publication of the algorithms. Features (table columns) are thematically grouped as described in Sec. 3. *(Column meanings:)* SND...is the algorithm sound? (Bullet **1**); COMPL...is it complete? (**2**); BEST-F...is it best-first? (**3**); MULT...is it multiple-solution? (**4**); CONF-DEP...is it conflict-dependent? (**5**); O-T-FLY...does it (if applicable) compute conflicts on-the-fly? (**6**); SEQ...is it sequential? (**7**); STATE...is it stateful? (**8**); GEN-APP...is it generally applicable? (**9**); BL-BOX-RS...is it black-box wrt. reasoning? (**10**); ANY-LOG...is it logics-agnostic? (**11**); POLY-SPACE...is it space-efficient? (**12**); *(Symbol meanings:)* ✓...yes; ∼...under certain circumstances; ×...no; ✓ (all)...all-complete; ✓ (p=$X$)...property-complete (wrt. property $X$); ✓ (one=$X$)...one-complete (wrt. property $X$); ✓ (gen)...generally best-first; ✓ (foc=$X$)...focused best-first (wrt. property $X$); ✓ (only=$X$)...only-best (wrt. property $X$); ✓ (bsno=$X$)...best-subset-no-order (wrt. property $X$); × (heur)...heuristic best-first; × (cp-b=$X$)...compilation-based (using target language $X$); × (dir)...direct; na...not applicable; × (circ)...specific to circuit diagnosis problems; × (bk)...uses bookkeeping mechanism for reasoning (ATMS for [13], HTMS for [10]); × (PL)...specific to propositional logic; × (PL/HL)...specific to propositional / Horn logic; ?...unknown. *(Table notes:)* (*)...algorithm name given in table not used in the original work; (1)...sound wrt. all diagnoses, but not wrt. all *minimal* diagnoses; (2)...unsound in most efficient configuration; can be parametrized to be sound; (3)...incomplete in most efficient configuration; can be parametrized to be complete; (4)...the diagnosis search is depth-first, i.e., requires linear space, but unclear if the used HTMS is polynomial space; (5)...polynomial-space hitting set computation, but assumes a given (i.e., preliminarily computed) set of conflicts; (6)...d-DNNF is less succinct (i.e., larger in general) than DNNF [7, Sec. 2], and compilation to DNNF may result in exponential-size compilations [6]; (7)...uses (linear-space) depth-first search, but unclear if FullReduce() function [10] (considers all conflicts at once) is polynomial space.

higher (empirical or theoretical) time complexity [50], or for a dropping of other desirable properties such as best-firstness [60, 20] or completeness [1]. *Relationship:* With the exception of one method [50], space-efficiency appears to be achievable only for algorithms that are not generally best-first (cf. Bullet **3**), or are preliminary wrt. conflict computation (cf. Bullet **6**) and whose complexity thus does not take into account the conflict generation phase.

**Remarks:**

- We do not propose a feature concerning the time complexity. This is due to well-known complexity results [5], which imply (unless P = NP) that there cannot be an algorithm that computes at least two minimal diagnoses and generally finishes in polynomial time; this holds even if reasoning (consistency checking) is in P, which however is already NP-complete if (only) propositional logic models are used (let alone more expressive logics, cf., e.g., [2]).
- The list of proposed features is not an exhaustive account of all possible properties diagnosis algorithms might have. There are further conceivable aspects from the *(a) theoretical viewpoint*, e.g., whether an algorithm uses abstractions or alterations of a DPI (cf. [52, Sec. 4]), or whether it is suitable for multi-observation problems (cf., e.g., [71]), *(b) empirical viewpoint*, e.g., whether an algorithm was experimentally evaluated, or which other methods an algorithm was compared against, *(c) presentation viewpoint*, e.g., whether formal proofs for algorithm properties are given, or from the *(d) pragmatic viewpoint*, e.g., whether there are freely accessible implementations of or tools based on an algorithm. Exploring further features like these is a future work topic.

## 4 Classification of Existing Works

Tab. 1 gives a classification of several existing works based on their characteristics wrt. the features suggested in Sec. 3:

- The table can be read row-wise to inspect the features of the diagnostic techniques, and column-wise to find methods with certain characteristics wrt. the features.
- We assessed the algorithms enumerated in the table as they are described in the respective cited work, without assuming any modifications or extensions.
- The list of algorithms studied in the table raises no claim to completeness. The idea is to illustrate the use(fulness) of the discussed features for algorithm assessment and comparison by showing the properties of a *significant number* of important methods in the literature. We plan to analyze further ones as part of our future work.

## 5 Conclusions

We propose a taxonomy for diagnosis algorithms, with the intention of helping researchers and practitioners in assessing, comparing, and selecting diagnostic techniques for their tasks and purposes. Specifically, we present a set of 12 crucial features of diagnosis techniques and classify 34 important existing methods based on these. In our study of the works in the literature, we observed that, for some algorithms, it was relatively hard to determine their properties wrt. the proposed taxonomy since various algorithmic aspects are often left implicit or not addressed at all. Moreover, even if properties are discussed, not all works provide formal proofs of them. Hence, we encourage authors to *explicitly* discuss the material properties of their proposed diagnostic approaches, for which we hope the suggested taxonomy will constitute a useful basis and guideline. Making algorithm characteristics explicit, clear and accessible will certainly help other researchers put novel works appropriately into the context of existing works and allow readers to better understand the proposed algorithms. And, adhering to shared assessment and categorization criteria while doing so can have several advantages. Examples are

- the better accessibility of diagnostic research through a reduction of potential ambivalence, e.g., resulting from different terminologies used in algorithm descriptions,
- a facilitation of comparisons between algorithms,
- a shared vocabulary to promote and simplify discussions among researchers,
- the usage of the taxonomy as a unified guiding principle for the structure of algorithmic analyses in papers (e.g., for theoretical studies of algorithms or related work sections),
- fair empirical evaluations that contrast methods which are actually comparable (e.g., comparing an incomplete method to a complete one wrt. performance might be pointless, as they simply accomplish different things and have different use cases),
- the potential inspiration for and detection of open research questions (e.g., find an algorithm with a particular subset of the features that no existing algorithm has),
- an easier, faster and more informed finding of a suitable algorithm for a specific purpose (e.g., is there a space-efficient method for a mobile device that is also sound, complete and best-first?),
- the better understanding of the evolution and reality in research and practice (e.g., that certain feature combinations are the reason why some techniques are not used in some application domains while they are state-of-the-art in others), or
- the realization that basically all algorithms have their right to exist, as they cover a wide variety of feature combinations and thus address a broad range of diagnostic problem scenarios, and that algorithms "superseding" others due to performance improvements mostly achieve this at the cost of losing some desirable properties.

Finally, due to the close relationship of diagnosis computation to other important domains such as hitting set problems [22, 45], abduction [5], set-theoretic duality [63], (MAX)SAT [8], MSMP [40], constraint satisfaction and optimization [15, 58], machine learning [26], or explainable AI [31], our study can have a positive impact far beyond the realm of diagnosis.

## Acknowledgements

This work was funded by the Austrian Science Fund (FWF), contract P-32445-N38.

## References

[1] Rui Abreu and Arjan van Gemund, 'A Low-Cost Approximate Minimal Hitting Set Algorithm and its Application to Model-Based Diagnosis', in *SARA*, (2009).
[2] Franz Baader, Diego Calvanese, Deborah L McGuinness, Daniele Nardi, and Peter F Patel-Schneider (eds.), 'The Description Logic Handbook', Cambridge University Press, (2007).
[3] Nuno Belard, Yannick Pencolé, and Michel Combacau, 'A theory of meta-diagnosis: reasoning about diagnostic systems', in *IJCAI*, (2011).
[4] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth, 'Occam's razor', *Inf. Process. Lett.*, **24**(6), 377–380, (1987).
[5] Tom Bylander, Dean Allemang, Michael Tanner, and John Josephson, 'The computational complexity of abduction', *AIJ*, **49**, 25–60, (1991).
[6] Adnan Darwiche, 'Decomposable negation normal form', *JACM*, **48**(4), 608–647, (2001).
[7] Adnan Darwiche, 'New advances in compiling CNF to decomposable negation normal form', in *ECAI*, (2004).
[8] Jessica Davies and Fahiem Bacchus, 'Solving MAXSAT by solving a sequence of simpler SAT instances', in *CP*, (2011).
[9] Johan de Kleer, 'Mininimum cardinality candidate generation', in *DX*, (2009).

[10] Johan de Kleer, 'Hitting set algorithms for model-based diagnosis', in *DX*, (2011).
[11] Johan de Kleer, Alan K Mackworth, and Raymond Reiter, 'Characterizing diagnoses and systems', *AIJ*, **56**, (1992).
[12] Johan de Kleer and Olivier Raiman, 'How to diagnose well with very little information', in *DX*, (1993).
[13] Johan de Kleer and Brian C Williams, 'Diagnosing multiple faults', *AIJ*, **32**(1), 97–130, (1987).
[14] Johan de Kleer and Brian C Williams, 'Diagnosis with behavioral modes', in *IJCAI*, (1989).
[15] Sharmi Dev Gupta, Begüm Genç, and Barry O'Sullivan, 'Explanation in constraint satisfaction: A survey', in *IJCAI*, (2021).
[16] Alexander Feldman, Gregory Provan, and Arjan van Gemund, 'Computing Minimal Diagnoses by Greedy Stochastic Search', in *AAAI*, (2008).
[17] Alexander Feldman, Ingo Pill, Franza Wotawa, Ion Matei, and Johan de Kleer, 'Efficient model-based diagnosis of sequential circuits', in *AAAI*, (2020).
[18] Alexander Feldman and Arjan van Gemund, 'A two-step hierarchical algorithm for model-based diagnosis', in *AAAI*, (2006).
[19] Alexander Felfernig, Gerhard Friedrich, and Erich Teppan, 'Automated Debugging and Repair of Utility Constraints in Recommender Knowledge Bases', in *DX*, (2007).
[20] Alexander Felfernig, Monika Schubert, and Christoph Zehentner, 'An efficient diagnosis algorithm for inconsistent constraint sets', *AI-EDAM*, **26**(1), 53–62, (2011).
[21] Gerhard Friedrich, Markus Stumptner, and Franz Wotawa, 'Model-based diagnosis of hardware designs', *AIJ*, **111**(1-2), 3–39, (1999).
[22] Andrew Gainer-Dewar and Paola Vera-Licona, 'The minimal hitting set generation problem: algorithms and computation', *SIAM J. on Discr. Math.*, **31**(1), 63–100, (2017).
[23] Rafael S Gonçalves, Samantha Bail, Ernesto Jiménez-Ruiz, Nicolas Matentzoglu, Bijan Parsia, Birte Glimm, and Yevgeny Kazakov, 'OWL reasoner evaluation (ORE) workshop 2013 results', in *ORE*, (2013).
[24] Dimitry Gorinevsky, Kevin Dittmar, Dinkar Mylaraswamy, and Emmanuel Nwadiogbu, 'Model-based diagnostics for an aircraft auxiliary power unit', in *ICCA*, (2002).
[25] Russell Greiner, Barbara A Smith, and Ralph W Wilkerson, 'A correction to the algorithm in Reiter's theory of diagnosis', *AIJ*, **41**(1), (1989).
[26] Dimitrios Gunopulos, Heikki Mannila, Roni Khardon, and Hannu Toivonen, 'Data mining, hypergraph transversals, and machine learning', in *SIGACT-SIGMOD-SIGART*, (1997).
[27] David Heckerman, John S Breese, and Koos Rommelse, 'Decision-theoretic troubleshooting', *Comm. of the ACM*, **38**(3), 49–57, (1995).
[28] Matthew Horridge, *Justification based Explanation in Ontologies*, PhD Thesis, Univ. of Manchester, 2011.
[29] Aimin Hou, 'A theory of measurement in diagnosis from first principles', *AIJ*, **65**(2), 281–328, (1994).
[30] John Hunt, 'Model-based software diagnosis', *Appl. Artif. Intell.*, **12**(4), 289–308, (1998).
[31] Alexey Ignatiev, Nina Narodytska, and Joao Marques-Silva, 'Abduction-based explanations for machine learning models', in *AAAI*, (2019).
[32] Dietmar Jannach and Thomas Schmitz, 'Model-based diagnosis of spreadsheet programs: a constraint-based debugging approach', *ASE*, **23**, 105–144, (2016).
[33] Dietmar Jannach, Thomas Schmitz, and Kostyantyn Shchekotykhin, 'Parallel model-based diagnosis on multi-core computers', *JAIR*, **55**, 835–887, (2016).
[34] Yunfei Jiang and Li Lin, 'Computing the minimal hitting sets with binary HS-tree', *J. of Softw.*, **13**(12), 2267–2274, (2002).
[35] Aditya Kalyanpur, *Debugging and Repair of OWL Ontologies*, PhD Thesis, Univ. of Maryland, College Park, 2006.
[36] Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and James Hendler, 'Debugging Unsatisfiable Classes in OWL Ontologies', *JWS*, **3**(4), 268–293, (2005).
[37] Johan de Kleer, 'Focusing on Probable Diagnoses', in *AAAI*, (1991).
[38] Lin Li and Jiang Yunfei, 'Computing minimal hitting sets with genetic algorithm', in *DX*, (2002).
[39] Li Lin and Yunfei Jiang, 'The computation of hitting sets: Review and new algorithms', *Inf. Process. Lett.*, **86**(4), 177–184, (2003).
[40] Joao Marques-Silva, Mikoláš Janota, and Anton Belov, 'Minimal sets over monotone predicates in boolean formulae', in *CAV*, (2013).
[41] Amit Metodi, Roni Stern, Meir Kalech, and Michael Codish, 'A novel SAT-based approach to model based diagnosis', *JAIR*, **51**, (2014).
[42] Ivan Mozetič, 'Hierarchical model-based diagnosis', *J. Man-Mach. Stud.*, (1991).
[43] Bijan Parsia, Evren Sirin, and Aditya Kalyanpur, 'Debugging OWL ontologies', in *WWW*, (2005).
[44] Ingo Pill and Thomas Quaritsch, 'Optimizations for the Boolean approach to computing minimal hitting sets', in *ECAI*, (2012).
[45] Raymond Reiter, 'A Theory of Diagnosis from First Principles', *AIJ*, **32**(1), 57–95, (1987).
[46] Patrick Rodler, Dietmar Jannach, Konstantin Schekotihin, and Philipp Fleiss, 'Are query-based ontology debuggers really helping knowledge engineers?', *KBS*, **179**, 92–107, (2019).
[47] Patrick Rodler, *Interactive Debugging of Knowledge Bases*, PhD Thesis, Univ. of Klagenfurt, 2015.
[48] Patrick Rodler, 'On Active Learning Strategies for Sequential Diagnosis', in *DX*, (2017).
[49] Patrick Rodler, 'Reuse, Reduce and Recycle: Optimizing Reiter's HS-Tree for Sequential Diagnosis', in *ECAI*, (2020).
[50] Patrick Rodler, 'Memory-limited model-based diagnosis', *AIJ*, **305**, 103681, (2022).
[51] Patrick Rodler, 'Random vs. Best-First: Impact of Sampling Strategies on Decision Making in Model-Based Diagnosis', in *AAAI*, (2022).
[52] Patrick Rodler, 'DynamicHS: Streamlining Reiter's hitting-set tree for sequential diagnosis', *Inf. Sci.*, **627**, 251–279, (2023).
[53] Patrick Rodler, 'Sequential Diagnosis by Systematic Search', *AIJ*, 103988, (2023).
[54] Patrick Rodler and Manuel Herold, 'StaticHS: A Variant of Reiter's Hitting Set Tree for Efficient Sequential Diagnosis', in *SoCS*, (2018).
[55] Patrick Rodler, Erich Teppan, and Dietmar Jannach, 'Randomized Problem-Relaxation Solving for Over-Constrained Schedules', in *KR*, (2021).
[56] Ana Armas Romero, Bernardo Cuenca Grau, and Ian Horrocks, 'MORe: Modular combination of OWL reasoners for ontology classification', in *ISWC*, (2012).
[57] Martin Sachenbacher, Andress Malik, and Peter Struss, 'From electrics to emissions: experiences in applying model-based diagnosis to real problems in real cars', in *DX*, (1998).
[58] Paul Saikko, *Implicit hitting set algorithms for constraint optimization*, Ph.D. dissertation, Univ. of Helsinki, 2019.
[59] Kostyantyn Shchekotykhin, Gerhard Friedrich, Philipp Fleiss, and Patrick Rodler, 'Interactive Ontology Debugging: Two Query Strategies for Efficient Fault Localization', *JWS*, **12-13**, 88–103, (2012).
[60] Kostyantyn Shchekotykhin, Gerhard Friedrich, Patrick Rodler, and Philipp Fleiss, 'Sequential diagnosis of high cardinality faults in knowledge-bases by direct diagnosis generation', in *ECAI*, (2014).
[61] Lei Shi and Xuan Cai, 'An Exact Fast Algorithm for Minimum Hitting Set', in *CSO*, (2010).
[62] Sajjad Siddiqi and Jinbo Huang, 'Sequential diagnosis by abstraction', *JAIR*, **41**, 329–365, (2011).
[63] John Slaney, 'Set-theoretic duality: A fundamental feature of combinatorial optimisation', in *ECAI*, (2014).
[64] Roni Stern, Meir Kalech, Alexander Feldman, and Gregory M Provan, 'Exploring the Duality in Conflict-Directed Model-Based Diagnosis', in *AAAI*, (2012).
[65] Pietro Torasso and Gianluca Torta, 'Model-based diagnosis through OBDD compilation: A complexity analysis', in *Reasoning, Action and Interaction in AI Theories and Systems*, (2006).
[66] Brian C Williams and Robert J Ragno, 'Conflict-directed A* and its role in model-based embedded systems', *Discr. Appl. Math.*, **155**(12), 1562–1595, (2007).
[67] Franz Wotawa, 'A variant of Reiter's hitting-set algorithm', *Inf. Process. Lett.*, **79**(1), 45–51, (2001).
[68] Zhao Xiangfu and Ouyang Dantong, 'A method of combining SE-tree to compute all minimal hitting sets', *Prog. Nat. Sci.*, **16**(2), 169–174, (2006).
[69] Safdar Zaman, Gerald Steinbauer, Johannes Maurer, Peter Lepej, and Suzana Uran, 'An integrated model-based diagnosis and repair architecture for ROS-based robot systems', in *ICRA*, (2013).
[70] Xiangfu Zhao and Dantong Ouyang, 'Deriving all minimal hitting sets based on join relation', *IEEE Trans. Syst. Man Cybern.*, **45**(7), 1063–1076, (2015).
[71] Huisi Zhou, Dantong Ouyang, Xiangfu Zhao, and Liming Zhang, 'Two Compacted Models for Efficient Model-Based Diagnosis', in *AAAI*, (2022).