Kalpa Publications in Computing
Volume 4, 2018, Pages 200–218
28th International Workshop on
Principles of Diagnosis (DX'17)

Inexpensive Cost-Optimized Measurement Proposal for Sequential Model-Based Diagnosis

Patrick Rodler[*], Wolfgang Schmid, and Konstantin Schekotihin

Alpen-Adria Universität, Klagenfurt, Austria
firstname.lastname@aau.at

Abstract

In this work we present strategies for (optimal) measurement computation and selection in model-based sequential diagnosis. In particular, assuming a set of leading diagnoses being given, we show how queries (sets of measurements) can be computed and optimized along two dimensions: expected number of queries and cost per query. By means of a suitable decoupling of two optimizations and a clever search space reduction the computations are done without any inference engine calls. For the full search space, we give a method requiring only a polynomial number of inferences and guaranteeing query properties existing methods do not provide. Evaluation results using real-world problems indicate that the new method computes (virtually) optimal queries instantly independently of the size and complexity of the considered diagnosis problems.

1 Introduction

Model-based diagnosis (MBD) is a widely applied approach to finding explanations, called *diagnoses*, for unexpected behavior of observed systems including hardware, software, knowledge bases, discrete event systems, feature models and user interfaces [27, 9, 40, 22, 25, 19, 11, 41]. In case the available observations about the system are insufficient for successful fault localization, i.e., multiple diagnoses exist, *sequential diagnosis (SQD)* methods collect additional information by generating a sequence of *queries* [7, 26, 10, 37, 35].[1] If chosen properly, each query's answer eliminates some diagnoses and thus reduces the diagnostic uncertainty. As query answering is often costly, the goal of SQD is to *minimize the diagnostic cost* in terms of, e.g., time or manpower required to achieve a diagnostic goal, e.g., a highly probable diagnosis. Because the problem of optimal query selection[2] is NP-complete [17], the cited SQD works minimize the *number of queries* by a greedy (e.g., one-step lookahead) measure m such as entropy [7]. However, they do not optimize the *query cost*, such as the time required to perform measurements [16].

[*] Corresponding author.

[1] Following the arguments of [26] we do not consider non-MBD sequential methods [24, 34, 43, 2, 15].

[2] Also known as *Optimal Test Sequencing Problem* [24] or *Optimal Decision Tree Problem* [17].

Contributions. We present a novel query optimization method that

1. is *logics- and reasoner-independent*, i.e. can handle system models formulated in any monotonic and decidable knowledge representation language and can incorporate any reasoner that is sound and complete for the used language. This allows for a general applicability to any MBD problem in the sense of [7, 27].

2. defines a query as a set of first-order sentences and thus generalizes the *measurement* notion of [7, 27].

3. can deal also with diagnosis problems where the query space is *implicit*.[3]

4. given a set of *leading diagnoses* [8], allows the two-dimensional optimization of the next query in terms of the *expected number of subsequent queries* (measure m) and *query cost* (measure c).

5. for an aptly refined (yet exponential) query search space, finds – *without any reasoner calls* – the *globally* optimal query wrt. measure c that *globally* optimizes measure m.[4]

6. for the full query search space, finds – with a polynomial number of reasoner calls – the (under reasonable assumptions) globally optimal query wrt. m that includes, if possible, only "cost-preferred" sentences (e.g., those answerable automatically using built-in sensors).

7. guarantees the proposal of queries that discriminate between *all* leading diagnoses and that *unambiguously identify the actual diagnosis*.

The efficiency of our approach is possible by the recognition that the optimizations of m and c can be decoupled and by using logical monotonicity as well as the inherent (already inferred) information in the ($\subseteq$-minimal) leading diagnoses. In particular, the method is inexpensive as it

(a) avoids the generation and examination of unnecessary (non-discriminating) or duplicate query candidates,

(b) *actually* computes only the *single* best query by its ability to estimate a query's quality without computing it, and

(c) guarantees soundness and completeness wrt. an exponential query search space independently of the properties and output of a reasoner.

Modern SQD methods like [7] and its derivatives [10, 35, 31] do not meet all properties (a) – (c). Moreover, by the generality of our query notion, our method explores a more complex search space than [7, 5], thereby guaranteeing property 7 above.

2 Preliminaries

2.1 Model-Based Diagnosis (MBD)

In this section we recapitulate important MBD concepts and draw on definitions of [27] to characterize a system and diagnoses. We will use the following notation throughout this work:

[3] We say that a diagnosis problem has an *implicit query space* iff all possible system measurements cannot be enumerated in polynomial time in the size of the system model. E.g., in a digital circuit all measurement points (and hence the possible queries) are given explicitly by the circuit's wires which can be directly extracted from the system description. By contrast, in e.g. knowledge base debugging problems the possible measurements (i.e. questions to an expert) must be (expensively) inferred and are not efficiently enumerable.

[4] The term *globally optimal* has its standard meaning (cf. [21, p. 184]) and emphasizes that the optimum over *all* queries in the respective query search space is meant.

SD	$\{\neg\mathrm{AB}(c_i) \to beh(c_i) \mid c_i \in \mathrm{COMPS}\}$	
COMPS	$\{c_1, c_2, c_3, c_4, c_5\}$	
normal behavior $beh(c_i)$ of components $c_i \in \mathrm{COMPS}$	$beh(c_1) : A \to B \land L$ $beh(c_3) : B \lor F \to H$ $beh(c_5) : \neg H \to G \land \neg A$	$beh(c_2) : A \to F$ $beh(c_4) : L \to H$
N	$n_1 : \{A \to H\}$	$\mathrm{OBS}, P = \emptyset$

Table 1: Running Example DPI Ex

Notation: If X is a collection of sets, then U_X and I_X denote the union and intersection of all elements of X, respectively. $K \models S$ for a set S is a shorthand for $K \models s$ for all $s \in S$. □

A *system* $(\mathrm{SD}, \mathrm{COMPS})$ consists of a set of components COMPS and a system description SD where $\{\neg\mathrm{AB}(c) \to beh(c) \mid c \in \mathrm{COMPS}\} \subseteq \mathrm{SD}$. The first-order sentence $beh(c)$ describes the normal behavior of c and AB is a distinguished abnormality predicate. Any behavior different from $beh(c)$ implies that component c is at fault, i.e., $\mathrm{AB}(c)$ holds.[5]

From the viewpoint of system diagnosis, evidence about the system behavior in terms of observations OBS, positive (P) and negative (N) measurements [27, 7, 12] is of interest.

Definition 1 (DPI)**.** *Let* COMPS *be a finite set of constants and let* SD*,* OBS*, each $p \in P$, and each $n \in N$ be finite sets of consistent first-order sentences. Then* $(\mathrm{SD}, \mathrm{COMPS}, \mathrm{OBS}, P, N)$ *is a* diagnosis problem instance (DPI).

Definition 2. *Let* $(\mathrm{SD}, \mathrm{COMPS}, \mathrm{OBS}, P, N)$ *be a DPI and* $\Delta \subseteq \mathrm{COMPS}$*. Then* $\mathrm{SD}^*[\Delta] := \mathrm{SD} \cup \mathrm{OBS} \cup U_P \cup \{\mathrm{AB}(c) \mid c \in \Delta\} \cup \{\neg\mathrm{AB}(c) \mid c \in \mathrm{COMPS} \setminus \Delta\}$ *denotes the behavior description of the system* $(\mathrm{SD}, \mathrm{COMPS})$ *given the observations* OBS *and the union of positive measurements* U_P *under the assumption that all components in* Δ *are faulty and all components in* $\mathrm{COMPS} \setminus \Delta$ *are healthy.*

The solutions of a DPI, i.e., the hypotheses that explain a given (faulty) system behavior, are called diagnoses:

Definition 3 (Diagnosis)**.** $\Delta \subseteq \mathrm{COMPS}$ *is a* diagnosis *for the DPI* $(\mathrm{SD}, \mathrm{COMPS}, \mathrm{OBS}, P, N)$ *iff* Δ *is* $\subseteq$*-minimal such that*

- $\mathrm{SD}^*[\Delta]$ *is consistent (*Δ *explains* OBS *and* P*), and*
- $\forall n \in N : \mathrm{SD}^*[\Delta] \not\models n$ *(*Δ *explains* N*).*

We denote the set of all diagnoses for a DPI X *by* $\mathfrak{D}_X$*.*

A diagnosis for a DPI exists iff $\mathrm{SD}^*[\mathrm{COMPS}] \not\models n$ for all $n \in N$ [13, Proposition 1].

Example 1: Consider the DPI Ex given by Tab. 1. Using, e.g., HS-TREE [27] we get (denoting components c_i by i) the set of all diagnoses $\mathfrak{D}_{\mathsf{Ex}} = \{\Delta_1, \Delta_2, \Delta_3\} = \{\{1,2,5\}, \{1,3,5\}, \{3,4,5\}\}$. E.g., $\Delta_2 \in \mathfrak{D}_{\mathsf{Ex}}$ due to Def. 3 and as $\mathrm{SD}^*[\Delta_2] = [\mathrm{SD} \cup \{\mathrm{AB}(c_1), \mathrm{AB}(c_3), \mathrm{AB}(c_5)\} \cup \{\neg\mathrm{AB}(c_2), \neg\mathrm{AB}(c_4)\}] \cup \mathrm{OBS} \cup U_P = [\{beh(c_2), beh(c_4)\}] \cup \emptyset \cup \emptyset = \{A \to F, L \to H\} \not\models \{A \to H\} = n_1 \in N$ and is consistent. □

[5] We make the *stationary health assumption* [10], i.e. the behavior of each $c \in \mathrm{COMPS}$ is constant during diagnosis.

2.2 Sequential Diagnosis (SQD)

Given multiple diagnoses for a DPI, SQD techniques extend the sets P and N by asking a user or an oracle (e.g., an automated system) to perform additional measurements in order to rule out irrelevant diagnoses. In line with the works of [33, 35, 28] we call a proposed measurement *query* and define it very generally as a set of first-order sentences (this subsumes the notion of measurement, e.g., in [7, 27]). The task of the oracle is to assess the correctness of the sentences in the query, thereby providing the required measurements. A query Q is *true* (t) if all sentences in Q are (necessarily) correct and *false* (f) otherwise.

Usually only a small computationally feasible set of *leading diagnoses* $\mathbf{D}$ (e.g., minimum cardinality [10] or most probable [4] ones) are exploited for measurement selection [8].

Any sets of diagnoses and first-order sentences satisfy:

Property 1. *Let X be a set of first-order sentences and $\mathbf{D} \subseteq \mathfrak{D}_{\mathsf{DPI}}$ for $\mathsf{DPI} = (\textsc{sd}, \textsc{comps}, \textsc{obs}, P, N)$. Then X induces a partition $\mathfrak{P}_{\mathbf{D}}(X) := \langle \mathbf{D}^+(X), \mathbf{D}^-(X), \mathbf{D}^0(X) \rangle$ on $\mathbf{D}$ where $\mathbf{D}^+(X) := \{\Delta \in \mathbf{D} \mid \textsc{sd}^*[\Delta] \models X\}$, $\mathbf{D}^-(X) := \{\Delta \in \mathbf{D} \mid \exists s \in N \cup \{\bot\} : \textsc{sd}^*[\Delta] \cup X \models s\}$ and $\mathbf{D}^0(X) = \mathbf{D} \setminus (\mathbf{D}^+(X) \cup \mathbf{D}^-(X))$.*

From a query, we postulate two properties, it must for any outcome **(1)** invalidate at least one diagnosis (*search space restriction*) and **(2)** preserve the validity of at least one diagnosis (*solution preservation*). In fact, the sets $\mathbf{D}^+(X)$ and $\mathbf{D}^-(X)$ are the key to deciding whether a set of sentences X is a query or not. Based on Property 1, we define:

Definition 4 (Query, q-Partition). *Let $\mathsf{DPI} = (\textsc{sd}, \textsc{comps}, \textsc{obs}, P, N)$, $\mathbf{D} \subseteq \mathfrak{D}_{\mathsf{DPI}}$ and Q be a set of first-order sentences with $\mathfrak{P}_{\mathbf{D}}(Q) = \langle \mathbf{D}^+(Q), \mathbf{D}^-(Q), \mathbf{D}^0(Q) \rangle$. Then Q is a* query *for $\mathbf{D}$ iff $Q \neq \emptyset$, $\mathbf{D}^+(Q) \neq \emptyset$ and $\mathbf{D}^-(Q) \neq \emptyset$. The set of all queries for $\mathbf{D}$ is denoted by $\mathbf{Q_D}$.*

$\mathfrak{P}_{\mathbf{D}}(Q)$ is called the q-partition (QP) *of Q iff Q is a query. Inversely, Q is called* a query with (or: for) the QP $\mathfrak{P}_{\mathbf{D}}(Q)$.

Given a QP $\mathfrak{P}$, we sometimes denote its three entries in turn $\mathbf{D}^+(\mathfrak{P})$, $\mathbf{D}^-(\mathfrak{P})$ and $\mathbf{D}^0(\mathfrak{P})$.

$\mathbf{D}^+(Q)$ and $\mathbf{D}^-(Q)$ denote those diagnoses in $\mathbf{D}$ consistent only with Q's positive and negative outcome, respectively, and $\mathbf{D}^0(Q)$ those consistent with both outcomes. Since $Q \in \mathbf{Q_D}$ implies that both $\mathbf{D}^+(Q)$ and $\mathbf{D}^-(Q)$ are non-empty, clearly Q's outcomes both dismiss and preserve at least one diagnosis. Note, in many cases a query also invalidates some (unknown) non-leading diagnoses $\mathfrak{D}_{\mathsf{DPI}} \setminus \mathbf{D}$.

The set $\mathbf{D}^0(Q)$ includes the so-called *uncommitted diagnoses* [7], i.e., those that cannot be eliminated given any outcome of Q. Hence, the size of the set $\mathbf{D}^0(Q)$ should be minimal, i.e., zero at best, for optimal diagnoses discrimination. The algorithm presented hereafter guarantees the computation of only Q's with $\mathbf{D}^0(Q) = \emptyset$. For example, the methods of [7, 35, 31] cannot ensure this property.

Example 1 (cont'd): Let $\mathbf{D} = \mathfrak{D}_{\mathsf{Ex}} = \{\Delta_1, \Delta_2, \Delta_3\}$. Then, $Q = \{F \to H\}$ is a query in $\mathbf{Q_D}$. To verify this, let us consider its QP $\mathfrak{P}_{\mathbf{D}}(Q) = \langle \{\Delta_1\}, \{\Delta_2, \Delta_3\}, \emptyset \rangle$. Since both $\mathbf{D}^+(Q)$ and $\mathbf{D}^-(Q)$ are non-empty, Q is in $\mathbf{Q_D}$. $\Delta_1 = \{1,2,5\} \in \mathbf{D}^+(Q)$ holds because $\textsc{sd}^*[\Delta_1] \models \{beh(c_3), beh(c_4)\} = \{B \lor F \to H, L \to H\}$ which in turn entails Q. On the other hand, e.g., $\Delta_2 = \{1,3,5\} \in \mathbf{D}^-(Q)$ since $\textsc{sd}^*[\Delta_2] \cup Q \models \{A \to F, L \to H, F \to H\} \models \{A \to H\} = n_1 \in N$. Hence, the outcome $Q = t$ implies that the diagnoses in $\mathbf{D}^-(Q) = \{\Delta_2, \Delta_3\}$ are invalidated, whereas $Q = f$ causes the dismissal of $\mathbf{D}^+(Q) = \{\Delta_1\}$. □

Applicability and diagnostic accuracy. For any non-singleton set of leading diagnoses, a discriminating query exists [28, Sec. 7.6]:

Property 2. $\forall \mathsf{DPI} : \mathbf{D} \subseteq \mathfrak{D}_{\mathsf{DPI}}, |\mathbf{D}| \geq 2 \implies \mathbf{Q}_{\mathbf{D}} \neq \emptyset$.

This has two implications: First, we need only precompute two diagnoses to generate a query and proceed with SQD. Despite its NP-completeness [3], the generation of two (or more) diagnoses is practical in many real-world settings [4, 36], making query-based SQD commonly applicable. Second, the query-based approach guarantees perfect diagnostic accuracy, i.e., the unambiguous identification of the actual diagnosis.

3 Query Optimization for Sequential Model-Based Diagnosis

3.1 Measurement Selection

As argued, the (q-)partition $\mathfrak{P}_{\mathbf{D}}(Q)$ enables both the *verification* whether a candidate Q is indeed a query and an *estimation of the impact* Q's outcomes have in terms of diagnoses invalidation. And, given (component) fault probabilities, it enables to *gauge the probability of observing a positive or negative query outcome* [7], e.g., to assess the uncertainty of Q. Active learning *query selection measures (QSMs)* $m : Q \mapsto m(Q) \in \mathbb{R}$ [33] use exactly these query properties characterized by the QP to assess how favorable a query is. They aim at selecting queries such that the expected number of queries until obtaining a deterministic diagnostic result is minimized, i.e.,

$$\sum_{\Delta \subseteq \textsc{comps}} p(\Delta) q_{\#}(\Delta) \;\rightarrow\; \min$$

where $p(\Delta)$ is the (a-priori) probability that $\{\textsc{ab}(c) \mid c \in \Delta\} \cup \{\neg\textsc{ab}(c) \mid c \in \textsc{comps} \setminus \Delta\}$ is the actual system state wrt. component functionality and $q_{\#}(\Delta)$ is the number of queries required, given the initial DPI, to derive that Δ must be the actual diagnosis. Solving this problem is known to be NP-complete as it amounts to optimal binary decision tree construction [17]. Hence, as it is common practice in SQD [7, 5, 26], we restrict our algorithm to the usage of QSMs that make a locally optimal query selection through a *one-step lookahead*. This has been shown to be optimal in many cases and nearly optimal in most cases [6]. Several different QSMs m such as *split-in-half*, *entropy*, or *risk-optimization* have been proposed, well studied and compared against one another other [7, 35, 31]. For instance, when using entropy as a QSM, m would be exactly the scoring function $\$()$ derived in [7]. Note, we assume w.l.o.g. that the optimal query wrt. any m is the one with *minimal* $m(Q)$.[6]

Besides trying to minimize the number of queries in a diagnostic session, a further goal can be the minimization of the query cost, e.g., in terms of time, manpower, difficulty or required tools. To this end, one can specify a *query cost measure (QCM)* $c : Q \mapsto c(Q) \in \mathbb{R}^+$. Examples of QCMs are $c_\Sigma(Q) := \sum_{i=1}^{k} c_i$ (prefer query with minimal overall cost, e.g., when c_i represents time) or $c_{\max}(Q) := \max_{i \in \{1,\dots,k\}} c_i$ (prefer query with minimal maximal cost of a single measurement, e.g., when c_i represents human cognitive load) where $Q = \{q_1, \dots, q_k\}$ and c_i is the cost of evaluating the truth of the first-order sentence q_i. The QCM $c_{|\cdot|}(Q) = |Q|$ is a special case of $c_\Sigma(Q)$ where $c_{|\cdot|}$ is equal to c_Σ in case $c_i = c_j$ for all i, j. Now, the problem addressed in this work is:

Problem 1 (Optimal Query Selection)**.**
Given: $\mathsf{DPI} := (\textsc{sd}, \textsc{comps}, \textsc{obs}, P, N)$*,* $\mathbf{D} \subseteq \mathfrak{D}_{\mathsf{DPI}}$ *with* $|\mathbf{D}| \geq 2$*, QSM* m*, QCM* c*, query search space* $\mathbf{S} \subseteq \mathbf{Q}_{\mathbf{D}}$.
Find: *A query* Q^* *with minimal cost wrt.* c *among all queries in* $\mathbf{S}$ *that are optimal wrt.* m*. Formally:* $Q^* = \arg\min_{Q \in \mathbf{OptQ}(m,\mathbf{S})} c(Q)$ *where* $\mathbf{OptQ}(m, \mathbf{S}) := \{Q' \mid Q' = \arg\min_{Q \in \mathbf{S}} m(Q)\}$.

[6] If for some QSM m the best query is the one which maximizes $m(Q)$, one can equivalently use the QSM $m'(Q) := -m(Q)$ which must be minimized.

Algorithm 1 Optimized Query Computation

```
Input: DPI $:= (\textsc{sd}, \textsc{comps}, \textsc{obs}, P, N)$, $\mathbf{D} \subseteq \mathfrak{D}_{\mathsf{DPI}}$, $|\mathbf{D}| \geq 2$, QSM $m$, QCM $c$, component fault probabilities
    $FP = \{p_i \mid p_i = p(c_i), c_i \in \textsc{comps}\}$, threshold $t_m$, sound and complete inference engine $Inf$, set $ET$ of
    entailment types, $pref$ (preference information used for query optimization), a Boolean enhance (if $true$,
    optional query enhancement is run)
Output: an optimized query $Q^* \in \mathbf{Q_D}$ wrt. $m$, $t_m$ and $c$ (cf. Theorems 2 and 3)
1: $\mathfrak{P} \leftarrow \textsc{optimizeQPartition}(\mathbf{D}, FP, m, t_m)$                          ▷ P1
2: if enhance $= true$ then                                                                          ▷ (optional)
3:     $Q' \leftarrow \textsc{expandQuery}(\mathsf{DPI}, \mathfrak{P}, ET, Inf)$                     ▷ P3
4:     $Q^* \leftarrow \textsc{optiMinimizeQuery}(\mathsf{DPI}, \mathfrak{P}, Q', pref, Inf)$        ▷ P3
5: else                                                                                              ▷ (default)
6:     $Q^* \leftarrow \textsc{optimizeQueryForQPartition}(\mathfrak{P}, c)$                         ▷ P2
7: return $Q^*$
```

Note, there can be multiple equally good queries $Q^* \in \mathbf{Q_D}$, i.e., the solution to Problem 1 might not be unique.

3.2 The Proposed Approach

The Algorithm[7] we propose to solve Problem 1 is given by Alg. 1. The described query computation procedure can be divided into three phases: P1 (line 1), P2 (line 6) and P3 (lines 3-4). Before we explain these phases in more detail we give an intuition and overview of the algorithm.

3.2.1 Overview and Intuition

In a nutshell, the presented query optimization method works as follows. In the first place, P1 optimizes the next query's discrimination properties (e.g. the expected information gain) based on the criteria imposed by the given QSM m, realized by a heuristic backtracking search. Then, as a first option, P2 computes an optimal query Q^* regarding the given QCM c by running a uniform-cost hitting set tree search over a suitable (and explicitly given) set of partial leading diagnoses. This is done in a way Q^* meets exactly the optimal discrimination properties determined in P1. P2 explores the largest possible query search space that can be handled without any reasoner calls in a complete way. The output Q^* suggests the inspection of the system component(s) that is least expensive for the oracle (QCM c) among all those that yield the highest information (QSM m). As a second option and alternative to P2, P3 performs a two-step optimization consisting of a first generalization of the addressed search space and a subsequent divide-and-conquer exploration of this search space focused on cost-preferred measurements. P3 returns a cost-optimal query Q^* (wrt. some QCM c) complying with the optimal discrimination properties fixed in P1. Q^* may include measurements of arbitrary type, depending on priorly definable requirements.

3.2.2 Phase P1

At this stage, we optimize the given QSM m – for now without regard to the QCM c, which is optimized later in P2 (default) or P3 (optional). This *decoupling of optimization steps* is possible since the QSM value $m(Q)$ of a query Q is only affected by the (unique) QP of Q and not by Q itself.[8] On the contrary,

[7] *In-depth* descriptions and discussions of all aspects of the proposed methods as well as proofs of all results are given in Sec. 3.4ff. of the extended version [29] of this paper. A Protégé [23] plugin implementing i.a. the presented algorithm can be found on http://isbi.aau.at/ontodebug/.

[8] Hence, we will sometimes write $m(\mathfrak{P})$ to denote the QSM-value of (any query for) the QP $\mathfrak{P}$.

the QCM value $c(Q)$ is a function of Q only and not of Q's QP. Therefore, the search performed in P1 will consider only QPs.

To verify whether a given 3-partition of $\mathbf{D}$ is a QP, however, we need a query Q for this QP which lets us determine whether $\mathbf{D}^+(Q) \neq \emptyset$ and $\mathbf{D}^-(Q) \neq \emptyset$ (cf. Def. 4). But:

Property 3. *For one query there is exactly one QP (immediate from Property 1). For one QP there might be multiple, in fact exponentially many queries (cf. Proposition 6 later).*

Therefore, we use the notion of a *canonical query (CQ)*, which is one well-defined query representative for a QP. From a CQ, we postulate *easiness of computation* and *exclusion of suboptimal QPs* with $\mathbf{D}^0 \neq \emptyset$ (cf. Sec. 2). The key to realizing these postulations is:

Definition 5. *Let* $X \subseteq \textsc{comps}$*. Then* $\textsc{beh}[X] := \{beh(c_i) \mid c_i \in X\}$.

The following property is immediate from Def. 2:

Property 4. $X \subseteq \textsc{comps} \Rightarrow \textsc{sd}^*[X] \models \textsc{beh}[\textsc{comps} \setminus X]$

From Property 1 and Def. 4 we can directly conclude:

Property 5. *A query* $Q \in \mathbf{Q_D}$ *is a subset of the common entailments of all KBs in the set* $\{\textsc{sd}^*[\Delta] \mid \Delta \in \mathbf{D}^+(Q)\}$.

Using Properties 4 and 5, the idea is now to restrict the space of entailments of the $\textsc{sd}^*[\cdot]$ KBs to the behavioral descriptions $beh(\cdot)$ of the system components. That is, each CQ should be some query $Q \subseteq \textsc{beh}[\textsc{comps}]$. In the light of Def. 4 and the $\subseteq$-minimality of diagnoses, necessary criteria for such queries are as follows:

Proposition 1. *[29, Cor. 19] Any query* $Q \subseteq \textsc{beh}[\textsc{comps}]$ *in* $\mathbf{Q_D}$ must *include some formulas in* $\textsc{beh}[U_\mathbf{D}]$, need not *include any formulas in* $\textsc{beh}[\textsc{comps} \setminus U_\mathbf{D}]$*, and* must not *include any formulas in* $\textsc{beh}[I_\mathbf{D}]$.[9]

Moreover, the deletion of any sentences in $\textsc{beh}[\textsc{comps} \setminus U_\mathbf{D}]$ *from* Q *does not alter the QP* $\mathfrak{P}_\mathbf{D}(Q)$.

Proposition 1 pinpoints the sentences crucial for the discrimination among the leading diagnoses $\mathbf{D}$. We give these sentences their own name:

Definition 6. $\mathsf{Disc}_\mathbf{D} := \textsc{beh}[U_\mathbf{D}] \setminus \textsc{beh}[I_\mathbf{D}] = \textsc{beh}[U_\mathbf{D} \setminus I_\mathbf{D}]$ *is called the* discrimination sentences wrt. $\mathbf{D}$.

CQs can now be characterized as follows:

Definition 7 (CQ)**.** *Let* $\emptyset \subset \mathbf{D}^+ \subset \mathbf{D}$*. Then* $Q_{\mathsf{can}}(\mathbf{D}^+) := \textsc{beh}[\textsc{comps} \setminus U_{\mathbf{D}^+}] \cap \mathsf{Disc}_\mathbf{D}$ *is* the canonical query (CQ) *wrt. seed* $\mathbf{D}^+$ *if* $Q_{\mathsf{can}}(\mathbf{D}^+) \neq \emptyset$*. Else,* $Q_{\mathsf{can}}(\mathbf{D}^+)$ *is undefined.*

Note, $\textsc{beh}[\textsc{comps} \setminus U_{\mathbf{D}^+}]$ are exactly the common $beh(\cdot)$ entailments of $\{\textsc{sd}^*[\Delta] \mid \Delta \in \mathbf{D}^+\}$ (cf. Property 5). The CQ extracts $\mathsf{Disc}_\mathbf{D}$ from these entailments, thereby removing all elements that do not affect the QP (cf. Proposition 1). By Defs. 4 and 7 and the $\subseteq$-minimality of diagnoses, we get:

Proposition 2. *[29, Proposition 50] If* Q *is a CQ, then* Q *is a query.*

The QP for a CQ is called canonical q-partition:

Definition 8 (CQP)**.** *A QP* $\mathfrak{P}'$ *for which a CQ* Q *exists with QP* $\mathfrak{P}'$*, i.e.,* $\mathfrak{P}_\mathbf{D}(Q) = \mathfrak{P}'$*, is called a* canonical QP (CQP).

[9] Please refer to the paragraph **Notation** in Sec. 2 for notation.

Since a CQ is a subset of $\textsc{beh}[\textsc{comps}]$ and diagnoses are $\subseteq$-minimal, we can derive:

Proposition 3. *[29, Proposition 52] Let $\mathfrak{P}$ be a CQP. Then $\mathbf{D}^0(\mathfrak{P}) = \emptyset$.*

Discussion: The restriction to CQs during P1 has some nice implications:

1. CQs can be generated by cheap set operations (*no inference engine calls*),
2. each CQ is a query in $\mathbf{Q}_{\mathbf{D}}$ for sure (Proposition 2), no verification of its QP (as per Def. 4) required, thence *no unnecessary* (non-query) *candidates generated*,
3. *automatic focus on favorable queries* wrt. the QSM m (those with empty $\mathbf{D}^0$, Proposition 3),
4. *no duplicate QPs generated* as there is a one-to-one relationship between CQs and CQPs (Property 3, Def. 7),
5. the explored search space for QPs is *not dependent on* the particular (entailments) output by an *inference engine*.

We emphasize that all these properties do **not** hold for normal (i.e., non-canonical) queries and QPs. The significant effect of these advantages of CQ(P)s will be demonstrated in Sec. 4. □

Example 1 (cont'd): Let $\mathbf{D}$ as before, $\mathsf{Disc}_{\mathbf{D}} = \textsc{beh}[U_{\mathbf{D}} \setminus I_{\mathbf{D}}] = \textsc{beh}[\{1,2,3,4,5\} \setminus \{5\}] = \textsc{beh}[\{1,2,3,4\}]$. Let us consider the seed $\mathbf{D}^+ = \{\Delta_1\} = \{\{1,2,5\}\}$. Then the CQ $Q_1 := Q_{\mathsf{can}}(\mathbf{D}^+) = (\textsc{beh}[\{1,2,3,4,5\} \setminus \{1,2,5\}]) \cap \textsc{beh}[\{1,2,3,4\}] = \textsc{beh}[\{3,4\}]$. The associated CQP is $\mathfrak{P}_1 = \langle \{\Delta_1\}, \{\Delta_2, \Delta_3\}, \emptyset \rangle$. Note, $\Delta \in \mathbf{D}^+(\mathfrak{P}_1)$ and $\Delta \in \mathbf{D}^-(\mathfrak{P}_1)$ hold for a $\Delta \in \mathbf{D}$ iff $\textsc{beh}[\textsc{comps} \setminus \Delta] \supseteq Q_1$ and $\textsc{beh}[\textsc{comps} \setminus \Delta] \not\supseteq Q_1$, respectively. E.g., $\Delta_3 \in \mathbf{D}^-(\mathfrak{P}_1)$ since $\textsc{beh}[\textsc{comps} \setminus \Delta_3] = \textsc{beh}[\{1,2\}] \not\supseteq \textsc{beh}[\{3,4\}] = Q_1$. That is, using CQs and CQPs, reasoning is traded for set operations and comparisons.

The seed $\mathbf{D}^+ = \{\Delta_1, \Delta_3\}$ yields $Q_2 := Q_{\mathsf{can}}(\mathbf{D}^+) = (\textsc{beh}[\{1,\dots,5\} \setminus \{1,\dots,5\}]) \cap \textsc{beh}[\{1,\dots,4\}] = \emptyset$, i.e., there is no CQ wrt. seed $\mathbf{D}^+$ and the partition $\langle \{\Delta_1, \Delta_3\}, \{\Delta_2\}, \emptyset \rangle$ with the seed $\mathbf{D}^+$ as first entry is no CQP. In fact, it is not a QP either. □

The q-partition search. Now, having at hand the notion of a CQP, we describe the (heuristic) search for an optimal CQP performed in P1.

A (heuristic) search problem [32] is defined by an *initial state*, a *successor function* enumerating all direct neighbor states of a state, the *step costs* from a state to a successor state, the *goal test* to determine if a given state is a goal state or not, (and some *heuristics* to estimate the remaining effort towards a goal state).

The type of the proposed search can be characterized as follows:

Depth-first, local best-first strategy At each state in the search tree, the best of all direct successor states is visited next. The goodness of states can be estimated by some heuristic function that assigns a real number to each state. The best state is the one with optimal (usually: minimal) heuristic value.

Backtracking strategy Given that all successors of a state have already been explored and no goal state has been found yet, the search backtracks and visits the next-best unexplored sibling of this state.

We define the initial state $\langle \mathbf{D}^+, \mathbf{D}^-, \mathbf{D}^0 \rangle$ as $\langle \emptyset, \mathbf{D}, \emptyset \rangle$. The idea is to transfer diagnoses step-by-step from $\mathbf{D}^-$ to $\mathbf{D}^+$ to construct all CQPs *systematically*. The step costs are irrelevant, only the found QP as such counts. Heuristics derived from the QSM m (cf., e.g., [35]) can be optionally integrated into the search to enable faster convergence to the optimum. A QP $\mathfrak{P}$ is a goal if it optimizes m up to the given threshold t_m (cf. [7]), i.e., if $|m(\mathfrak{P}) - m_{opt}| \leq t_m$ where m_{opt} denotes the (theoretically) optimal value of m.[10] In order to characterize a suitable successor function, we define a direct neighbor of a QP as follows:

Definition 9. *Let* $\mathfrak{P}_i := \langle \mathbf{D}_i^+, \mathbf{D}_i^-, \emptyset \rangle$*,* $\mathfrak{P}_j := \langle \mathbf{D}_j^+, \mathbf{D}_j^-, \emptyset \rangle$ *be partitions of* $\mathbf{D}$*. Then,* $\mathfrak{P}_i \mapsto \mathfrak{P}_j$ *is a* minimal $\mathbf{D}^+$-transformation *from* $\mathfrak{P}_i$ *to* $\mathfrak{P}_j$ *iff* $\mathfrak{P}_j$ *is a CQP,* $\mathbf{D}_i^+ \subset \mathbf{D}_j^+$ *and there is no CQP* $\langle \mathbf{D}_k^+, \mathbf{D}_k^-, \emptyset \rangle$ *with* $\mathbf{D}_i^+ \subset \mathbf{D}_k^+ \subset \mathbf{D}_j^+$.

A CQP $\mathfrak{P}'$ *is called a* successor *of a partition* $\mathfrak{P}$ *iff* $\mathfrak{P}'$ *results from* $\mathfrak{P}$ *by a minimal* $\mathbf{D}^+$*-transformation.*

Intuitively, a successor $\mathfrak{P}'$ of a partition $\mathfrak{P}$ is the result of transferring a $\subseteq$-minimal set of diagnoses from $\mathbf{D}^-(\mathfrak{P})$ to $\mathbf{D}^+(\mathfrak{P})$ such that $\mathfrak{P}'$ is a CQP.

For the successors of the initial state we get:

Proposition 4. *[29, Proposition 55] The CQPs* $\langle \{\Delta\}, \mathbf{D} \setminus \{\Delta\}, \emptyset \rangle$ *for* $\Delta \in \mathbf{D}$ *are exactly all successors of* $\langle \emptyset, \mathbf{D}, \emptyset \rangle$.

To specify the successors of an intermediate CQP $\mathfrak{P}_k$ in the search, we draw on diagnoses' *traits*:

Definition 10. *Let* $\mathfrak{P}_k = \langle \mathbf{D}_k^+, \mathbf{D}_k^-, \emptyset \rangle$ *be a CQP and* $\Delta_i \in \mathbf{D}_k^-$*. Then the* trait $\Delta_i^{(k)}$ *of* Δ_i *is defined as* $\text{BEH}[\Delta_i \setminus U_{\mathbf{D}_k^+}]$.

The relation $\sim_k$ associating two diagnoses in $\mathbf{D}_k^-$ iff their trait is equal is obviously an equivalence relation. Now, Defs. 7, 8 and 9 let us derive:

Proposition 5. *[29, Cor. 25] Let* $EC := \{E_1, \dots, E_s\}$ *be the set of all equivalence classes wrt.* $\sim_k$*.* $\mathfrak{P}_k$ *has successors iff* $s \geq 2$*. In this case, all successors are given by* $\langle \mathbf{D}_k^+ \cup E, \mathbf{D}_k^- \setminus E, \emptyset \rangle$ *where* $E \in EC$ *and* E *has a* $\subseteq$*-minimal trait among all classes* $E' \in EC$.

By Def. 9 which requires both *minimal changes* between state and successor state and *the latter to be a CQP*, we have:

Theorem 1. *Usage of the successor function as given in Proposition 4 (for initial state) and Proposition 5 (for intermediate states) makes the search for CQPs sound and complete.*

For the number of CQ(P)s, we obtain the following result:

Proposition 6. *[29, Cor. 23] Let* $\mathbf{CQP_D}$ *denote the set of CQPs for diagnoses* $\mathbf{D}$ *with* $|\mathbf{D}| \geq 2$*. Then* $|\mathbf{CQP_D}| = |\{U_{\mathbf{D}^+} \mid \emptyset \subset \mathbf{D}^+ \subset \mathbf{D}, U_{\mathbf{D}^+} \neq U_{\mathbf{D}}\}| \geq |\mathbf{D}|$.

Whether QPs $\langle \mathbf{D}^+, \mathbf{D}^-, \emptyset \rangle$ exist which are no CQPs is not yet clarified, but both theoretical (cf. [29, Sec. 3.4.2]) and empirical evidence indicate the negative. E.g., an analysis of $\approx 900\,000$ QPs we ran for different diagnoses sets $\mathbf{D}$ for different DPIs showed that *all QPs were indeed CQPs*. And, in all evaluated cases (see Sec. 4) optimal CQPs wrt. all QSMs m given in diagnosis literature [7, 35, 31] were found. Hence:

Conjecture 1. *Let* $(\mathbf{C})\mathbf{QP_D}$ *denote the sets of (C)QPs (all with* $\mathbf{D}^0 = \emptyset$*) for diagnoses* $\mathbf{D}$*. Then* $\mathbf{CQP_D} = \mathbf{QP_D}$.

[10] I.e., m_{opt} is the best value regarding the QSM m that any (fictitious) QP might achieve. There need not be a query in $\mathbf{Q_D}$ for which m evaluates to m_{opt}.

Example 1 (cont'd): Reconsider the CQP $\mathfrak{P}_1 = \langle\{\Delta_1\}, \{\Delta_2, \Delta_3\}, \emptyset\rangle$. The traits are $\Delta_2^{(1)} = \textsc{beh}[\{1,3,5\} \setminus \{1,2,5\}] = \textsc{beh}[\{3\}]$ and $\Delta_3^{(1)} = \textsc{beh}[\{3,4\}]$, representing two equivalence classes wrt. $\sim_1$. There is only one class with $\subseteq$-minimal trait, i.e., $\{\Delta_2\}$. Hence, there is just a single successor CQP $\mathfrak{P}_2 = \langle\{\Delta_1, \Delta_2\}, \{\Delta_3\}, \emptyset\rangle$ of $\mathfrak{P}_1$ (cf. Proposition 5). Recall, we argued that $\langle\{\Delta_1, \Delta_3\}, \{\Delta_2\}, \emptyset\rangle$ is indeed no (C)QP. By Proposition 6, there are $|\{\{1,2,5\}, \{1,3,5\}, \{3,4,5\}, \{1,2,3,5\}, \{1,3,4,5\}\}| = 5$ different CQPs wrt. $\mathbf{D}$. Note, Conjecture 1 is true here, i.e., the $\mathbf{CQP_D}$ search is complete wrt. $\mathbf{QP_D}$. □

The next example shows one entire execution of the CQP search performed by phase P1 of Alg. 1:

Example 2: Consider the leading diagnoses $\mathbf{D}$ given by[11]

$$\{\Delta_1, \dots, \Delta_6\} = \{\{2,3\}, \{2,5\}, \{2,6\}, \{2,7\}, \{1,4,7\}, \{3,4,7\}\}$$

Further on, let the diagnoses probabilities

$$\langle p(\Delta_1), \dots, p(\Delta_6)\rangle = \langle 0.01, 0.33, 0.14, 0.07, 0.41, 0.04\rangle$$

The search tree for a goal CQP wrt. the entropy QSM $m := \$$ (see [7, p. 11]) and $t_m := 0.01$ produced by P1 is shown in Fig. 1. Roughly, m evaluates a QP $\mathfrak{P}$ the better, the lower the difference between the probabilities $p(\mathbf{D}^+(\mathfrak{P})) := \sum_{\Delta\in\mathbf{D}^+(\mathfrak{P})} p(\Delta)$ and $p(\mathbf{D}^-(\mathfrak{P})) := \sum_{\Delta\in\mathbf{D}^-(\mathfrak{P})} p(\Delta)$ (cf. [35]). Let us therefore assume a very simple heuristic function h that assigns $h(\mathfrak{P}) = |p(\mathbf{D}^+(\mathfrak{P})) - 0.5|$ to a QP $\mathfrak{P}$ where smaller h values imply more promising QPs wrt. m to visit next. Here, the optimal QSM-value $m_{opt} = 0$ and thus a CQP $\mathfrak{P}$ is a goal state iff $|m(\mathfrak{P})| \leq t_m$ (cf. Alg. 1).

In Fig. 1, a node in the search tree representing the CQP $\mathfrak{P}_k = \left\langle \mathbf{D}_k^+, \mathbf{D}_k^-, \mathbf{D}_k^0 \right\rangle$ is denoted by a frame including a table with three rows where (1) the topmost row shows $p(\mathbf{D}_k^+) \mid p(\mathbf{D}_k^-)$ (relevant for computing m and h), (2) the middle row depicts $\mathbf{D}_k^+ \mid \mathbf{D}_k^-$ and (3) the bottommost row gives $U_{\mathbf{D}_k^+} \mid \{\Delta_i^{(k)} \mid \Delta_i \in \mathbf{D}_k^-\}$ (cf. Def. 10).[12] The framed value at the bottom right corner of the large frame quotes the heuristic value $h(\mathfrak{P}_k)$. No such value for the root node is given since it is not a QP and hence does not qualify as a solution. Furthermore, the (for CQPs) always empty $\mathbf{D}_k^0$ set is omitted. A frame is dashed / continuous / double if the associated node is generated (but not expanded) / expanded / a returned goal CQP. Arrows represent minimal $\mathbf{D}^+$-transformations, i.e., an arrow's destination QP is a result of a minimal $\mathbf{D}^+$-transformation applied to its source (q-)partition. Arrow labels give the set of diagnoses and the probability mass moved from the $\mathbf{D}^-$-set of the source (q-)partition to the $\mathbf{D}^+$-set of the destination QP.

Starting from the root node representing the partition $\langle\emptyset, \mathbf{D}, \emptyset\rangle$, the successor function generates all possible CQPs resulting from the transfer of a single diagnosis from the $\mathbf{D}^-$-set of the initial state to its $\mathbf{D}^+$-set. Since there are six diagnoses in $\mathbf{D}$, the initial state has exactly six successors (see Proposition 4). The best successor $\mathfrak{P}_1 := \langle\{\Delta_5\}, \{\Delta_1, \Delta_2, \Delta_3, \Delta_4, \Delta_6\}, \emptyset\rangle$ with minimal $h(\mathfrak{P}_1) = 0.09$ is selected for expansion.

For $\mathfrak{P}_1$, the successor function generates exactly two CQPs that result from it by a minimal $\mathbf{D}^+$-transformation (cf. Proposition 5). This can be seen by considering the traits of the diagnoses in $\mathbf{D}^-(\mathfrak{P}_1)$ shown in the right column of the third row in the table representing $\mathfrak{P}_1$. Among the five traits there are only two $\subseteq$-minimal ones, i.e., $\Delta_4^{(1)} := \{2\}$ and $\Delta_6^{(1)} := \{3\}$. All the other traits are proper supersets of either of these. This means that all successors of $\mathfrak{P}_1$ can be constructed by shifting either

[11] Note, since P1, as argued, works on the basis of set operations and without reasoning at all, the actual DPI giving rise to these diagnoses is not relevant in this example.

[12] For simplicity, we will write just the set X to represent a trait $\Delta_i^{(k)} = \textsc{beh}[X]$.

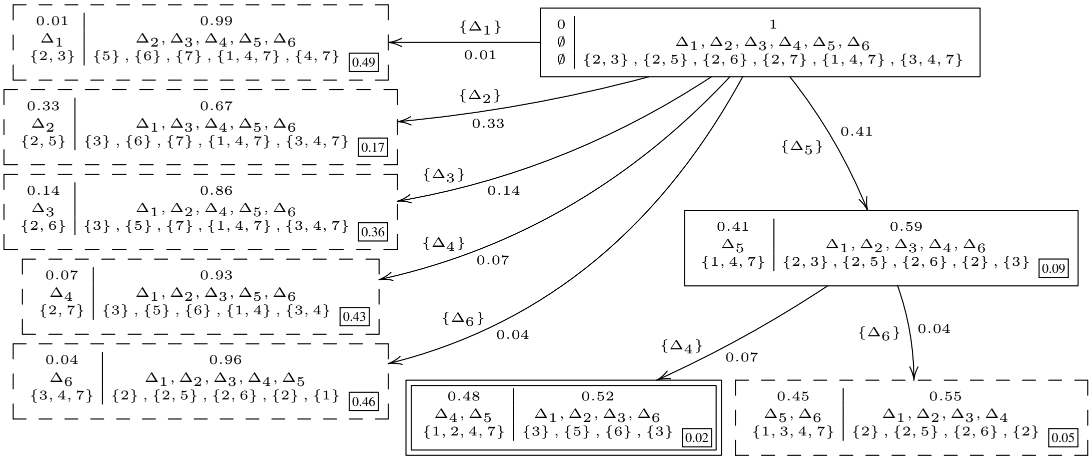


Figure 1: Search for an optimal QP in phase P1 of Alg. 1 using the entropy QSM m and threshold $t_m := 0.01$.

Δ_4 or Δ_6 from $\mathbf{D}^-(\mathfrak{P}_1)$ to $\mathbf{D}^+(\mathfrak{P}_1)$ yielding $\mathfrak{P}_{21} := \langle\{\Delta_4, \Delta_5\}, \{\Delta_1, \Delta_2, \Delta_3, \Delta_6\}, \emptyset\rangle$ and $\mathfrak{P}_{22} := \langle\{\Delta_5, \Delta_6\}, \{\Delta_1, \Delta_2, \Delta_3, \Delta_4\}, \emptyset\rangle$, respectively.

At this stage, the best QP among the two successors $\mathfrak{P}_{21}$ and $\mathfrak{P}_{22}$ of $\mathfrak{P}_1$ (depth-first, *local* best-first) is determined for expansion by means of h. As $p(\mathbf{D}^+(\mathfrak{P}_{21}))$ differs by less (0.02) from 0.5 than $p(\mathbf{D}^+(\mathfrak{P}_{22}))$ (0.05), $\mathfrak{P}_{21}$ is chosen. However, as t_m has been set to 0.01 and $m(\mathfrak{P}_{21}) \approx 0.001 \leq 0.01$, $\mathfrak{P}_{21}$ is a goal and returned as the solution of phase P1 of Alg. 1. Note, there were no backtrackings as the used heuristic h guided the search directly towards a goal state. Such behavior could also be frequently observed in our experiments (see Sec. 4). □

3.2.3 Phase P2

Phase P1 returns an optimal (C)QP $\mathfrak{P}_k$ wrt. the QSM m. Property 3 indicates that there might be still a large search space for an optimal query wrt. the QCM c for this QP. The task in P2 is to find such query efficiently.

From $\mathfrak{P}_k$, we can obtain the associated CQ Q_k (as per Def. 7). However, usually a least requirement of any QCM c is i.a. the $\subseteq$-minimality of a query to avoid unnecessary measurements. To this end, let $\mathsf{Tr}(\mathfrak{P}_k)$ denote the set of all $\subseteq$-minimal traits wrt. $\sim_k$. Given a collection of sets $X = \{x_1, \dots, x_n\}$, a set $H \subseteq U_X$ is a *hitting set (HS)* of X iff $H \cap x_i \neq \emptyset$ for all $x_i \in X$. Then:

Proposition 7. *[29, Proposition 61] $Q \subseteq \mathsf{Disc}_{\mathbf{D}}$ is a $\subseteq$-minimal query with QP $\mathfrak{P}_k$ iff $Q = H$ for some $\subseteq$-minimal HS H of $\mathsf{Tr}(\mathfrak{P}_k)$.*

Hence, all $\subseteq$-minimal reductions of CQ Q_k under preservation of the (already fixed and optimal) QP $\mathfrak{P}_k$ can be computed, e.g., using the classical HS-TREE [27]. However, there is a crucial difference to standard application scenarios of HS-TREE, namely the fact that all sets to label the tree nodes (i.e., the $\subseteq$-minimal traits) are readily available (without further computations). Consequently, the construction of the tree runs swiftly, as our evaluation will confirm. Note also, in principle we only require a *single* minimal hitting set, i.e., query. Moreover, HS-TREE can be used as uniform-cost (UC) search (cf., e.g., [28, Chap. 4]), incorporating the QCM c to find queries in best-first order wrt. c. In fact, all QCMs (i.e., c_Σ, $c_{\max}$, $c_{|\cdot|}$) discussed above can be optimized using UC HS-TREE. In case some QCM c is not suitable for UC search, a brute force HS-TREE search over all $\subseteq$-minimal queries is often practical as well (no expensive operations involved), as our evaluations (cf. Sec. 4) have revealed. Hence, P1 and P2 provide a solution to Problem 1 *without a single inference engine call.*

Theorem 2. *P1 and P2 compute a solution Q^* to Problem 1 where* $\mathbf{S} := \{\textsc{beh}[X] \mid X \subseteq \textsc{comps}\}$.

Example 1 (cont'd): Recall the CQP $\mathfrak{P}_1$ and let the QCM be $c := c_{|\cdot|}$ (i.e., queries of minimum cardinality should be preferred). Then $\mathsf{Tr}(\mathfrak{P}_1) = \{\textsc{beh}[\{3\}]\}$, i.e., by Proposition 7 there is a single c-optimal query $\textsc{beh}[\{3\}]$ for $\mathfrak{P}_1$, a proper subset of the CQ $\textsc{beh}[\{3, 4\}]$ for $\mathfrak{P}_1$. Considering the CQP $\mathfrak{P}_3 := \langle \{\Delta_2\}, \{\Delta_1, \Delta_3\}, \emptyset \rangle$, $\mathsf{Tr}(\mathfrak{P}_3) = \{\textsc{beh}[\{2\}], \textsc{beh}[\{4\}]\}$ and thus we have (Proposition 7) a single c-optimal query $\textsc{beh}[\{2, 4\}]$ which happens to be equal to the CQ for $\mathfrak{P}_3$. □

So, *optionally*, by selecting $\mathsf{enhance} := \mathit{false}$ (cf. Alg. 1), the query Q^* optimized along two dimensions (number of queries, cost per query) over the restricted search space $\mathbf{S}$ (Theorem 2) is directly proposed as next measurement. A $\textsc{beh}[\cdot]$ query like Q^* corresponds to a direct examination of one or more system components. This could mean to, e.g., ping servers in a distributed system [2], test gates using a voltmeter in circuits [7] or ask stakeholders of a software / configuration / KB system about the correctness of code lines / constraints / logical sentences [42, 12, 13].

Alternatively, the already optimal CQP $\mathfrak{P}_k$ returned by P1 can be exploited to build a solution query to Problem 1 with full search space $\mathbf{S} = \mathbf{Q_D}$. To this end, $\mathsf{enhance}$ is set to true, causing the execution of phase P3 instead of P2.

3.2.4 Phase P3

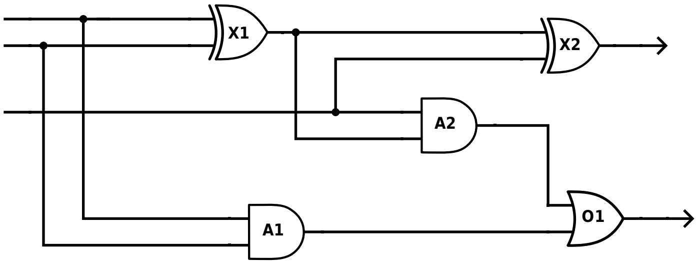
In this phase, first, using the CQ Q_k of $\mathfrak{P}_k$, a (finite) set Q_{exp} of first-order sentences of types ET (e.g., atoms or sentences of type $A \to B$) are computed. Q_{exp} must meet:

1. $\textsc{sd}^*[X] \models Q_{\mathsf{exp}}$ where X is some (superset of a) diagnosis such that $Q_k \subseteq \textsc{sd}^*[X]$ (*entailed by a consistent system behavior KB*),
2. no $q_i \in Q_{\mathsf{exp}}$ is an entailment of $\textsc{sd}^*[X] \setminus Q_k$ (logical dependence on Q_k, *no irrelevant sentences*) and
3. the expansion of Q_k by Q_{exp} does *not alter the (already fixed and optimal) QP* $\mathfrak{P}_k$, i.e., $\mathfrak{P}_k = \mathfrak{P}_{\mathbf{D}}(Q_k \cup Q_{\mathsf{exp}})$.

Proposition 8. *(cf. [29, Proposition 62]) Let $Ent_{ET}(X)$ be a* monotonic *consequence operator realized by some inference engine that computes a finite set of entailments of types ET of a KB X. Postulations 1 – 3 above are satisfied if* $Q_{\mathsf{exp}} := Ent_{ET}(\textsc{sd}^*[U_{\mathbf{D}}] \cup Q_k) \setminus Ent_{ET}(\textsc{sd}^*[U_{\mathbf{D}}])$.

Finally, the expanded query $Q' := Q_k \cup Q_{\mathsf{exp}}$ can be minimized to get a $\subseteq$-minimal subset of it under preservation of the associated QP $\mathfrak{P}_k$. For this purpose, one can use a variant of the polynomial divide-and-conquer method QUICKXPLAIN [18], e.g., the MINQ procedure given in [28, p.111ff.]. However, we propose to alter the input to MINQ as follows: Assume that Q' can be partitioned into a subset of *cost-preferred sentences* $Q'_{\mathsf{c}+}$ (e.g., those measurements executable automatically by available built-in sensors) and cost-dispreferred ones $Q'_{\mathsf{c}-} = Q' \setminus Q'_{\mathsf{c}+}$ (e.g., manual measurements), as specified by the parameter *pref* (see Alg. 1). Let the input to MINQ be the list $Q'_{\mathsf{c}+} \| asc(Q'_{\mathsf{c}-})$ (reordering of Q') where $asc(Q'_{\mathsf{c}-})$ means that $Q'_{\mathsf{c}-}$ is sorted in ascending order by sentence cost and $\|$ denotes the standard concatenation operator. Then:

Proposition 9. *(cf. [29, Cor. 27])* MINQ *with input $Q'_{\mathsf{c}+} \| asc(Q'_{\mathsf{c}-})$ returns a $\subseteq$-minimal query $Q^* \subseteq Q'$ such that $\mathfrak{P}_{\mathbf{D}}(Q^*) = \mathfrak{P}_k$. Further, if such a query comprising only $Q'_{\mathsf{c}+}$ (and no $Q'_{\mathsf{c}-}$) sentences exists, then $Q^* \subseteq Q'_{\mathsf{c}+}$. Else, Q^* optimizes the QCM $c_{\max}$ (see page 3) among all $\subseteq$-minimal subsets of Q' with QP $\mathfrak{P}_k$.*



circuit inputs (from top to bottom)
1
0
1
circuit outputs (from top to bottom)
1
0

Figure 2: Full adder circuit (left) and related observations (right).

Note, phase P3, i.e., query expansion (Proposition 8) together with optimized minimization (Proposition 9), requires only a *polynomial number of inference engine calls* [18].

Theorem 3. *Let Conjecture 1 hold and the QCM be $c_{\max}$ (see page 3). Then P3, exploiting the QP output by P1 along with Proposition 8 and 9, solves Problem 1 with full search space $\mathbf{S} = \mathbf{Q_D}$ (under the postulated exclusion of suboptimal queries Q with $\mathbf{D}^0(Q) \neq \emptyset$).*

Note, Conjecture 1 is by no means necessary for the proper functioning of our presented algorithms. In case Conjecture 1 turned out to be wrong, the consequence would be just the invalidity of *perfect* completeness wrt. all QPs achieved by the restriction to only CQPs. Still, we could cope well with that since CQs and CQPs bring along very nice computational properties (see the discussion above), CQP search spaces are nevertheless of substantial size (see Sec. 4), CQPs prove extremely efficient by the total avoidance of reasoning (see Sec. 4) and enable to find optimal queries wrt. negligible thresholds t_m for all tested cases and QSMs (see Sec. 4).

Example 1 (cont'd): Assume the QP $\mathfrak{P}_1$ is returned by P1. Let the cost c_i of a sentence q_i be the number of literals in its clausal form. As shown before, the CQ of $\mathfrak{P}_1$ is $Q_1 := \textsc{beh}[\{3,4\}] = \{B \lor F \to H, L \to H\}$. Using Proposition 8 with ET set to "definite clauses with singleton body", we get $Q_{\mathsf{exp}} = Ent_{ET}(Q_1) \setminus Ent_{ET}(\emptyset) = \{B \to H, F \to H, L \to H\}$. So, $Q' = \{B \to H, F \to H, L \to H, B \lor F \to H\}$. Suppose ET defines exactly the cost-preferred sentences, i.e., $Q'_{c+} = Q_{\mathsf{exp}}$. Running $\textsc{minQ}$ with input $Q_{\mathsf{exp}} \| \{B \lor F \to H\}$ yields $Q^* = \{F \to H\}$, a query including only cost-preferred elements (cf. Proposition 9). It is easily verified by Property 1 that Q^* has still the QP $\mathfrak{P}_1$. □

3.2.5 Recapitulation

To conclude Sec. 3, let us illustrate the entire query selection process executed by Alg. 1 drawing on the full adder circuit example [14, 27] depicted by Fig. 2.

Example 3: The DPI resulting from the circuit in Fig. 2 is $\mathsf{Circ} := (\textsc{sd}, \textsc{comps}, \textsc{obs}, P, N)$ where $\textsc{comps}$ includes the five gates X_1, X_2 (xor), A_1, A_2 (and), O_1 (or), and $\textsc{sd}$ includes one sentence $\neg\textsc{ab}(c) \to beh(c)$ for each $c \in \textsc{comps}$ and models the wires as connections between output and input terminals of gates, e.g., $out(X_1) = in1(X_2)$ for the wire connecting X_1 and X_2. At this, $beh(c)$ describes the nominal behavior of a gate, e.g., $beh(X_1) := out(X_1) = xor(in1(X_1), in2(X_1))$ for the xor-gate X_1 where $in1(.), in2(.)$ denote the first and second input of a gate, $out(.)$ its output, and $xor(.)$ the standard xor-function (cf. [27, p. 4]). Below we denote the set of sentences describing the circuit's wires by $\textsc{wires}$. Further, $\textsc{obs} = \{in1(X_1) = 1, in2(X_1) = 0, in1(A_2) = 1, out(X_2) = 1, out(O_1) = 0\}$ includes the observations about the circuit's in- and outputs (see Fig. 2). The collections of positive and negative measurements, P and N, are initially empty.

Suppose we got the information from the manufacturer of the gates that and-, or- and xor-gates fail with a probability of 0.05, 0.02 and 0.01, respectively (input FP to Alg. 1). As discussed in [27, p. 7], the set of all diagnoses for Circ is $\mathfrak{D}_{\mathsf{Circ}} = \{\Delta_1, \Delta_2, \Delta_3\} = \{\{X_1\}, \{X_2, A_2\}, \{X_2, O_1\}\}$. Let the leading diagnoses be $\mathbf{D} := \mathfrak{D}_{\mathsf{Circ}}$. Appealing to the formula given in [7, Sec. 4.4], the diagnoses probabilities (normalized over $\mathbf{D}$ and rounded) resulting from FP amount to $\langle p(\Delta_1), p(\Delta_2), p(\Delta_3)\rangle = \langle 0.93, 0.05, 0.02\rangle$.

(Phase P1:) Assuming the same QSM m, threshold t_m and heuristic h as in Example 2, P1 runs a brute force search over all (C)QPs as there is no QP that qualifies as a goal wrt. t_m. In this case, P1 outputs the best QP visited throughout its execution, namely $\mathfrak{P}_1 := \langle \{\Delta_1\}, \{\Delta_2, \Delta_3\}, \emptyset\rangle$ with $h(\mathfrak{P}_1) = 0.43$ and $m(\mathfrak{P}_1) \approx 0.63$.

(Phase P2:) Let us suppose that a globally optimal query wrt. the QCM c_Σ (see page 3) over the restricted search space considered by P2 (see Theorem 2) is desired by the user, i.e., $\mathsf{enhance} := \mathit{false}$. And, let the expected cost of testing an and-, or- and xor-gate, respectively, be 1, 3 and 2. Then $\mathsf{Tr}(\mathfrak{P}_1) = \{\Delta_2^{(1)}, \Delta_3^{(1)}\} = \{\textsc{beh}[\{X_2, A_2\}], \textsc{beh}[\{X_2, O_1\}]\}$ is used to extract the c_Σ-optimal query $Q^* = \textsc{beh}[\{X_2\}] = \{out(X_2) = xor(in1(X_2), in2(X_2))\}$ as the minimal HS with least cost ($c_\Sigma(Q^*) = 2$) of all elements of $\mathsf{Tr}(\mathfrak{P}_1)$ (cf. Proposition 7). Note, the (only) other $\subseteq$-minimal HS is $Q_{alt1} := \textsc{beh}[\{A_2, O_1\}]$ with a cost of $c_\Sigma(Q_{alt1}) = 1 + 3 = 4$. Q^* corresponds to the question "Does gate X_2 work properly?".

(Phase P3:) Given that a query optimized over the full search space (see Theorem 3) is wanted, $\mathsf{enhance} := \mathit{true}$ causes the execution of phase P3 (instead of phase P2). As an input Inf to Alg. 1 we assume, e.g., some constraint propagator, similar to the one described in [7], which computes predictions of the values at the circuit's wires. Moreover, we suppose that the preferred entailment types ET are exactly those stating values of wires, e.g., $out(A_1) = 1$.

As a first step in P3, the CQ Q of $\mathfrak{P}_1$ is computed as per Def. 7 and 8 as $Q := \textsc{beh}[\textsc{comps} \setminus U_{\mathbf{D}^+(\mathfrak{P}_1)}] \cap \textsc{beh}[U_\mathbf{D} \setminus I_\mathbf{D}] = \textsc{beh}[\{X_1, X_2, A_1, A_2, O_1\} \setminus \{X_1\}] \cap \textsc{beh}[\{X_1, X_2, A_2, O_1\} \setminus \emptyset] = \textsc{beh}[\{X_2, A_2, O_1\}]$. Then, Q is used to compute the query expansion Q_{exp} according to Proposition 8 as $Q_{\mathsf{exp}} = \mathit{Ent}_{ET}(\textsc{sd}^*[U_\mathbf{D}] \cup Q) \setminus \mathit{Ent}_{ET}(\textsc{sd}^*[U_\mathbf{D}])$ which is by Def. 2 equal to $\mathit{Ent}_{ET}(\textsc{beh}[\{A_1\}] \cup \textsc{wires} \cup \textsc{obs} \cup \emptyset \cup \textsc{beh}[\{X_2, A_2, O_1\}]) \setminus \mathit{Ent}_{ET}(\textsc{beh}[\{A_1\}] \cup \textsc{wires} \cup \textsc{obs} \cup \emptyset)$. After reasoning, the latter is equal to $\{out(X_1) = 0, out(A_2) = 0, out(A_1) = 0\} \setminus \{out(A_1) = 0\} = \{out(X_1) = 0, out(A_2) = 0\}$.

Next, the contraction of the expanded query $Q' = Q \cup Q_{\mathsf{exp}} = \{beh(X_2), beh(A_2), beh(O_1), out(X_1) = 0, out(A_2) = 0\}$ takes place. Let us assume that a user wants to avoid component inspections, i.e., the query should not include any $beh(.)$ sentences. This is reflected by specifying pref (cf. Alg. 1) so as to $Q'_+ := Q_{\mathsf{exp}}$. Finally, the list $Q'_+ \| asc(Q'_-) = [out(X_1) = 0, out(A_2) = 0, beh(A_2), beh(X_2), beh(O_1)]$ is passed to $\textsc{minQ}$ (regarding $asc()$, recall the gate inspection costs mentioned in P2 above). The resulting output is the optimized contracted query $Q^* = \{out(X_1) = 0\}$ (with QP $\mathfrak{P}_1$). Indeed, Q^* includes only cost-preferred elements. We note that Q^* is the only $\subseteq$-minimal query with QP $\mathfrak{P}_1$ which is a subset of Q'_+. For the only other $\subseteq$-minimal query comprising only elements from Q'_+ is $Q_{alt2} := \{out(A_2) = 0\}$ which has not the QP $\mathfrak{P}_1$, i.e. is not QP-preserving. The actual QP $\mathfrak{P}_\mathbf{D}(Q_{alt2})$ of Q_{alt2} is $\langle \{\Delta_1, \Delta_2\}, \{\Delta_3\}, \emptyset\rangle$.

Hence, Alg. 1 suggests to probe the wire connecting gate X_1 with gates X_2 and A_2. Taking into account the query outcome probabilities estimated from the diagnoses probabilities (cf. [7, Sec. 4.3]), we see that there is a strong bias (probability 0.93) towards a measurement outcome of $out(X_1) = 0$.

In this case, as indicated by the QP $\mathfrak{P}_1$ of Q^*, only a single measurement is needed to single out Δ_1 as the actual diagnosis, i.e., to come to the conclusion that X_1 must be faulty. □

4 Evaluation

4.1 Experimental Settings and Dataset

To evaluate our method, we used real-world inconsistent knowledge-based (KB) systems as **(1)** they pose a hard challenge for query selection methods due to the implicit nature of the possible queries (must be derived by inference; not directly given such as wires in a circuit), **(2)** any MBD system in the sense of [27] is described by a KB, **(3)** the *type* of the underlying system is irrelevant to our method, only its *size* and *(reasoning) complexity* – for the optional P3 – and the *DPI structure*, e.g., size, number or probability of diagnoses – for P1, P2 – are critical. To account for this, we used systems (Tab. 2, col. 1) of different size (Tab. 2, col. 2), complexity (Tab. 2, col. 3) and DPI structure (Tab. 2, col. 4).

For each KB $\mathcal{K}$ in col. 1 of Tab. 2, a DPI was built as $(\textsc{sd}, \textsc{comps}, \textsc{obs}, P, N)$ where $\textsc{comps}$ included one component c_i for each logical axiom $a_i \in \mathcal{K}$, $\textsc{sd} = \{\neg\textsc{ab}(c_i) \rightarrow \mathit{beh}(c_i) \mid 1 \leq i \leq |\mathcal{K}|\}$ where $\mathit{beh}(c_i) := a_i$ for $1 \leq i \leq |\mathcal{K}|$, $\textsc{obs} = \emptyset$ (note, $\textsc{sd}$ under the assumption of no faulty components is inconsistent, without any observations, as $\mathcal{K}$ is inconsistent), $P = \emptyset$ and $N = \emptyset$.

In our experiments,[13] for each DPI Sys in Tab. 2 and each $n \in \{10, 20, \dots, 80\}$, we randomly generated 5 different $\mathbf{D} \in \mathfrak{D}_{\mathsf{Sys}}$ with $|\mathbf{D}| = n$ using INV-HS-TREE [36] with randomly shuffled input. Each $\Delta \in \mathbf{D}$ was assigned a uniformly random probability.

For each of these 5 $\mathbf{D}$-sets, we used (a) entropy (ENT) [7] and (b) split-in-half (SPL) [35] as QSM m and $c_{|.|}$ (cf. page 3) as QCM c, and then ran phases (i) P1+P2 and (ii) P3 to compute an optimized query as per Theorems 2 and 3, respectively. We set the optimality threshold t_m to 0.01 in (a) and 0 in (b). The search in P1 used the same heuristic as stated in Example 2 for ENT and $h(\mathfrak{P}) = \big||\mathbf{D}^+(\mathfrak{P})| - \frac{1}{2}|\mathbf{D}|\big|$ for SPL, both drawing on ideas from [20] that were already exploited in [35]. In P3, *pref* was chosen to assign all simple definite clauses of the form $\forall x(A(x) \rightarrow B(x))$ and all facts of the form $A(a)$ to Q'_+.

4.2 Experimental Results

The obtained experimental results are shown in Fig. 3.[14] Times for SPL are omitted for clarity as they were quasi the same as for ENT. The dark gray area shows the # of CQPs addressed by P1, and the light gray line the time for P1+P2 using ENT. It is evident that P1+P2 always finished in less than 0.03 sec outputting an optimized query wrt. m and c. Note, albeit P1+P2 solve Prob. 1 for a restricted search space $\mathbf{S}$ (cf. Theorem 2), $|\mathbf{CQP_D}|$, a fraction of $|\mathbf{S}|$, already averaged to, e.g., 300 (over $|\mathbf{D}| = 10$ cases) and $> 530\,000$ ($|\mathbf{D}| = 80$). That $|\mathbf{S}|$ is sufficiently large for all sizes $|\mathbf{D}|$ is also substantiated by the fact that *in each single run* an optimal query wrt. the very small t_m ($\frac{1}{10}$ of t_m used in [35]) was found in $\mathbf{S}$. Also, a brute force (BF) search (dashed line) iterating over all possible CQPs is feasible in most cases – finishing within 1 min for all runs (up to search space sizes $> 120\,000$) except the $|\mathbf{D}| \geq 30$ cases for system CE (where up to 3 million CQPs were computed). This extreme speed is possible due to the *complete avoidance of costly reasoner calls*. The optional further query enhancement in P3 using the DL reasoner Pellet [38] computing entailments of types $ET := \{\text{classification}, \text{realization}\}$ [1] always finished within 4 sec and returned the globally optimal query wrt. QCM $c_{\max}$ (Theorem 3). The median output query size after P1+P2+P3 was 3.4. In additional scalability tests using $|\mathbf{D}| = 500$ for the large enough DPIs (CC, CE, T, E), P1+P2 always ended in < 0.6 sec, P3 in < 40 sec.

[13] All tests were run on a Core i7 with 3.4 GHz, 16 GB RAM and Windows 7 64-bit OS.

[14] More comprehensive results of these and further experiments are discussed in-depth in [30, Sec. 4].

System	$\|$COMPS$\|$	Complexity [a]	#D/min/max [b]
University (U) [c]	49	$\mathcal{SOIN}^{(D)}$	90/3/4
MiniTambis (M) [c]	173	$\mathcal{ALCN}$	48/3/3
CMT-Conftool (CC) [d]	458	$\mathcal{SIN}^{(D)}$	934/2/16
Conftool-EKAW (CE) [d]	491	$\mathcal{SHIN}^{(D)}$	953/3/10
Transportation (T) [c]	1300	$\mathcal{ALCH}^{(D)}$	1782/6/9
Economy (E) [c]	1781	$\mathcal{ALCH}^{(D)}$	864/4/8
Opengalen-no-propchains (O) [e]	9664	$\mathcal{ALEHIF}^{(D)}$	110/2/6
Cton (C) [e]	33203	$\mathcal{SHF}$	15/1/5

[a] Description Logic expressivity, cf. [1, p. 525ff.].

[b] #D, min, max denote #, min. and max. size of all diagnoses (computable in ≤ 8 h).

[c] Sufficiently complex systems (#D $\geq$ 40) used in [35].

[d] Hardest diagnosis problems mentioned in [39].

[e] Hardest diagnosis problems tested in [35].

Table 2: Systems used in the experiments

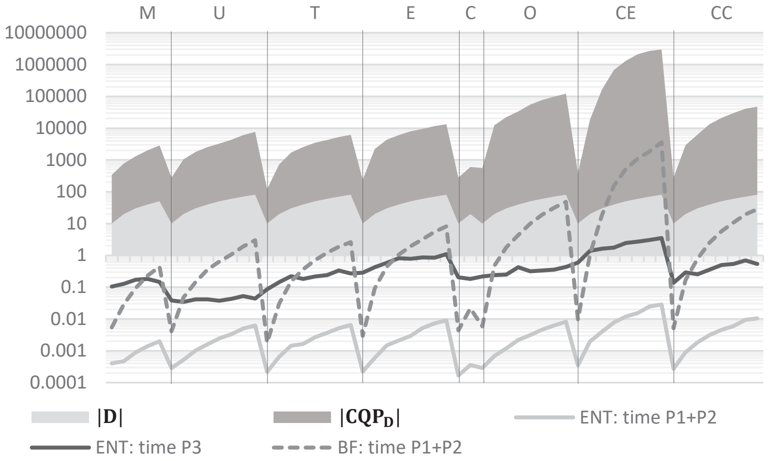


Figure 3: Results for systems in Tab. 2 (x-axis): # of leading diagnoses $|\mathbf{D}|$, associated size $|\mathbf{CQP_D}|$ of CQP search space, and computation time (sec) required by phases P1+P2 and P3 for QSM ENT with threshold $t_m = 0.01$ and brute force (BF) search (y-axis).

We also simulated P1 by a method using *non-canonical* QPs, thus relying on a reasoner. For *no* DPI in Tab. 2 a result for $|\mathbf{D}| > 15$ could be found *within an hour*. And, the quality of the returned QP (if any) wrt. m was *never* better than for P1. So, not even in a single tested case, a query quality loss was given due to the restriction to our *canonical* concepts.

5 Conclusions

We present a search that addresses the optimal measurement (*query*) computation and selection problem for sequential model-based diagnosis. The method is independent of the (monotonic and decidable) knowledge representation language describing the system model and of the used (sound and complete) inference engine. Therefore, it is generally applicable across different application domains to any model-based diagnosis problem conforming to [7, 27]. Moreover, the approach can efficiently deal with hard sequential diagnosis cases involving implicit[15] and large-sized query spaces.

In particular, we allow a query to be optimized along two dimensions, i.e., number of queries and cost per query. We show that the optimizations of these properties can be decoupled and considered in sequence. For a suitably restricted (still exponential) query search space (very close approximations of) *global optima* wrt. given query quality measures are found *without any calls to an inference engine* in *negligible time* for diagnosis problems of any size and complexity (given the precomputation of ≥ 2 diagnoses is feasible). For instance, query search spaces of size up to 3 million can be handled *instantaneously* (< 0.1 sec). For the full search space, under reasonable assumptions, the *globally optimal* query wrt. a cost-preference measure can be found within 4 sec for up to 80 leading diagnoses. Scalability tests involving a vast number of 500 precomputed diagnoses (search space size in $O(2^{500})$ only in the first computation phase of the algorithm) always output an optimized query in less than 0.7 sec (restricted search space) and 40 sec (full search space), respectively.

Acknowledgments

This work was supported by the Carinthian Science Fund (KWF), contract KWF-3520/26767/38701.

References

[1] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2007.

[2] Mark Brodie, Irina Rish, Sheng Ma, and Natalia Odintsova. Active probing strategies for problem diagnosis in distributed systems. In *IJCAI'03*, pages 1337–1338, 2003.

[3] Tom Bylander, Dean Allemang, Michael Tanner, and John Josephson. The computational complexity of abduction. *Artificial Intelligence*, 49:25–60, 1991.

[4] Johan de Kleer. Focusing on probable diagnoses. In *AAAI'91*, pages 842–848, 1991.

[5] Johan de Kleer and Olivier Raiman. How to diagnose well with very little information. In *Working Notes of the Fourth International Workshop on Principles of Diagnosis*, pages 160–165, 1993.

[6] Johan de Kleer, Olivier Raiman, and Mark Shirley. One step lookahead is pretty good. In *Readings in model-based diagnosis*, pages 138–142, 1992.

[7] Johan de Kleer and Brian C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32(1):97–130, 1987.

[8] Johan de Kleer and Brian C. Williams. Diagnosis with behavioral modes. In *IJCAI'89*, pages 1324–1330, 1989.

[9] Oskar Dressler and Peter Struss. The consistency-based approach to automated diagnosis of devices. *Principles of Knowledge Representation*, pages 269–314, 1996.

[10] Alexander Feldman, Gregory M. Provan, and Arjan J. C. van Gemund. A model-based active testing approach to sequential diagnosis. *Journal of Artificial Intelligence Research*, 39:301–334, 2010.

[11] Alexander Felfernig, Gerhard Friedrich, Karl Isak, Kostyantyn Shchekotykhin, Erich Teppan, and Dietmar Jannach. Automated debugging of recommender user interface descriptions. *Applied Intelligence*, 31(1):1–14, 2009.

[12] Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, and Markus Stumptner. Consistency-based diagnosis of configuration knowledge bases. *Artificial Intelligence*, 152(2):213–234, 2004.

[13] Gerhard Friedrich and Kostyantyn Shchekotykhin. A General Diagnosis Method for Ontologies. In *ISWC'05*, pages 232–246, 2005.

[15] See footnote 3.

[14] Michael R. Genesereth. The use of design descriptions in automated diagnosis. *Artificial Intelligence*, 24(1-3):411–436, 1984.

[15] Alberto Gonzalez-Sanchez, Rui Abreu, Hans-Gerhard Gross, and Arjan J.C. van Gemund. Spectrum-based sequential diagnosis. In *AAAI'11*, pages 189–196, 2011.

[16] David Heckerman, John S. Breese, and Koos Rommelse. Decision-theoretic troubleshooting. *Communications of the ACM*, 38(3):49–57, 1995.

[17] Laurent Hyafil and Ronald L. Rivest. Constructing optimal binary decision trees is NP-complete. *Information processing letters*, 5(1):15–17, 1976.

[18] Ulrich Junker. QUICKXPLAIN: Preferred Explanations and Relaxations for Over-Constrained Problems. In *AAAI'04*, volume 3, pages 167–172, 2004.

[19] Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding all Justifications of OWL DL Entailments. In *ISWC'07*, pages 267–280, 2007.

[20] Richard E. Korf. A complete anytime algorithm for number partitioning. *Artificial Intelligence*, 106(2):181–203, December 1998.

[21] David G. Luenberger and Yinyu Ye. *Linear and Nonlinear Programming*. Springer Publishing Company, Inc., 2015.

[22] Cristinel Mateis, Markus Stumptner, Dominik Wieland, and Franz Wotawa. Model-Based Debugging of Java Programs. In *AADEBUG'00*, 2000.

[23] Natalya F. Noy, Michael Sintek, Stefan Decker, Monica Crubézy, Ray W. Fergerson, and Mark A. Musen. Creating Semantic Web Contents with Protégé-2000. *IEEE Intelligent Systems*, 16(2):60–71, 2000.

[24] Krishna R. Pattipati and Mark G. Alexandridis. Application of heuristic search and information theory to sequential fault diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(4):872–887, 1990.

[25] Yannick Pencolé and Marie-Odile Cordier. A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks. *Artificial Intelligence*, 164(1):121–170, 2005.

[26] Jurryt Pietersma, Arjan J.C. van Gemund, and Andre Bos. A model-based approach to sequential fault diagnosis. In *IEEE Autotestcon, 2005*, pages 621–627, 2005.

[27] Raymond Reiter. A Theory of Diagnosis from First Principles. *Artificial Intelligence*, 32(1):57–95, 1987.

[28] Patrick Rodler. *Interactive Debugging of Knowledge Bases*. PhD thesis, Alpen-Adria Universität Klagenfurt, 2015. http://arxiv.org/pdf/1605.05950v1.pdf.

[29] Patrick Rodler. Towards better response times and higher-quality queries in interactive knowledge base debugging. Technical report, Alpen-Adria Universität Klagenfurt, 2016. http://arxiv.org/pdf/1609.02584v2.pdf.

[30] Patrick Rodler, Wolfgang Schmid, and Konstantin Schekotihin. A generally applicable, highly scalable measurement computation and optimization approach to sequential model-based diagnosis. Technical report, Alpen-Adria Universität Klagenfurt, 2017. https://arxiv.org/pdf/1711.05508.pdf.

[31] Patrick Rodler, Kostyantyn Shchekotykhin, Philipp Fleiss, and Gerhard Friedrich. RIO: Minimizing User Interaction in Ontology Debugging. In *Web Reasoning and Rule Systems*, pages 153–167. 2013.

[32] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 3rd edition, 2010.

[33] Burr Settles. *Active Learning*. Morgan and Claypool Publishers, 2012.

[34] Mojdeh Shakeri, Vijay Raghavan, Krishna R. Pattipati, and Ann Patterson-Hine. Sequential testing algorithms for multiple fault diagnosis. *IEEE Trans. Systems, Man, and Cybernetics, Part A*, 30(1):1–14, 2000.

[35] Kostyantyn Shchekotykhin, Gerhard Friedrich, Philipp Fleiss, and Patrick Rodler. Interactive Ontology Debugging: Two Query Strategies for Efficient Fault Localization. *Web Semantics: Science, Services and Agents on the World Wide Web*, 12-13:88–103, 2012.

[36] Kostyantyn Shchekotykhin, Gerhard Friedrich, Patrick Rodler, and Philipp Fleiss. Sequential diagnosis of high cardinality faults in knowledge-bases by direct diagnosis generation. In *ECAI'14*, pages 813–818, 2014.

[37] Sajjad Siddiqi and Jinbo Huang. Sequential diagnosis by abstraction. *Journal of Artificial Intelligence Re-*

search, 41:329–365, 2011.

[38] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):51–53, 2007.

[39] Heiner Stuckenschmidt. Debugging OWL Ontologies - A Reality Check. In *Proceedings of the 6th International Workshop on Evaluation of Ontology-based Tools and the Semantic Web Service Challenge (EON)*, pages 1–12, 2008.

[40] Markus Stumptner and Franz Wotawa. Debugging functional programs. In *IJCAI'99*, pages 1074–1079, 1999.

[41] Jules White, David Benavides, Douglas C. Schmidt, Pablo Trinidad, Brian Dougherty, and Antonio Ruiz Cortés. Automated diagnosis of feature model configurations. *Journal of Systems and Software*, 83(7):1094–1107, 2010.

[42] Franz Wotawa. On the relationship between model-based debugging and program slicing. *Artificial Intelligence*, 135(1-2):125–143, 2002.

[43] Alenka Zuzek, Anton Biasizzo, and Franc Novak. Sequential diagnosis tool. *Microprocessors and Microsystems - Embedded Hardware Design*, 24(4):191–197, 2000.