Generating Diagnoses from Conflicting Component Sets with Continuous Extents

Emmanuel Benazera and **Louise Travé-Massuyès**
LAAS-CNRS, Université de Toulouse, 7 av. du Colonel Roche,
31077 Toulouse Cedex 4, France. email: ebenazer,louise@laas.fr

Abstract

Many techniques in model-based diagnosis and other research fields find the hitting sets of a collection of sets. Existing techniques apply to sets of finite elements only. This paper addresses the computation of the hitting sets of a collection of sets whose elements have convex or non-convex, bounded or unbounded continuous attributes, also referred to as continuous extents. We assume the conflict sets are known and we present a novel procedure, the Continuous Hitting Set algorithm (CHS) for transforming conflict sets of elements with continuous extends into minimal hitting sets.

1 Introduction

Many theoretical and practical problems can partly reduce to an instance of the minimal hitting set problem, or its close variant the minimum set covering problem. One widely recognized application is in the field of model-based diagnosis [Reiter, 1987]. In this approach, a system is a tuple $(COMPS, SD, OBS)$; $COMPS$ is a finite set of system components; SD is the system description; OBS is the observation. A diagnosis is a minimal set $D \subseteq COMPS$ such that under the assumption that all other components are behaving correctly, D *explains* the observation given SD. In the diagnosis community, D is said to be consistent with SD and OBS. This approach to diagnosis has two steps: (i) a collection of all minimal conflict sets is computed; (ii) the conflict sets are transformed into diagnoses. A conflict set $s \subseteq COMPS$ is such that the assumption that all components in s are behaving correctly is not consistent with SD and OBS. A minimal conflict set is such that it does not contain any other conflict set. Reiter showed that the minimal diagnoses are the minimal hitting sets of the collection of minimal conflicts.

Since the beginning of model-based diagnosis, several algorithms for computing the hitting sets have been introduced. Most rely on the building of a so-called HS-DAG [Greiner *et al.*, 1989] or HS-tree [Wotawa, 2001] but other representations exist [Haenni, 1997; Lin and Jian, 2003]. All these techniques transform conflict sets of discrete elements into diagnoses. But in many applications of model-based diagnosis, the conflicts contain more information. This information includes but is not limited to intervals of possible failure time in systems with functional delays, or continuous parameter ranges found in fault models. For example, in systems with delays, several conflicts may involve the same components with different estimates of the symptom occurence dates [Travé-Massuyès and Calderon-Espinoza, 2007].

In this paper, we address step (ii) and assume all minimal conflicts are given. Each conflict element is a component with associated bounded or unbounded intervals over a continuous line. We assume there is a single continuous attribute per component, but this assumption has no incidence on the generality of the presented method. The problem with conflicting component sets with continuous attributes is that minimal diagnoses are conditioned upon the continuous values. This is because a minimal diagnosis corresponds to a minimal continuous region. A diagnosis in this context is a set of k components along with a set of bounded or unbounded regions of $\Re^k$. Existing algorithms are not designed to find and construct these regions. A naive strategy would be to apply these algorithms to a collection of conflicts with selected elements of the continuous lines. However, since the hitting set problem has a worst case performance that is exponential the size of the collection of conflicts, this would hardly prove an efficient approach. Moreover, many points that belong to the same minimal diagnoses would be computed independently.

This paper presents a general computational method for finding the hitting sets of a collection of conflicts with continuous attributes. The algorithm is named CHS for Continuous Hitting Sets. Starting from the classical approach, the proposed solution searches the hitting sets in an aggregate space of diagnoses. Similarly to the classical methods the CHS has both an expansion and a pruning phases. It is shown how the pruning phase dominates the computational effort. Simulation experiments on hundreds of randomly generated conflicts assess the main properties and the scalability of the CHS.

2 Problem Definition

2.1 Conflicts and diagnoses

We consider a tuple $(COMPS, SD, OBS)$. $COMPS$ is the set of physical components composing the physical system. OBS is a set of observations. Given OBS, the diagnosis operation derives all sets of faulty components of $COMPS$ that may explain the facts. In the sense of Reiter, the conflict sets, or conflicts for short, are the sets of components which cannot behave normally altogether according to the observations. A minimal conflict is a conflict that does not strictly include (in the sense of set inclusion) any conflict. [Reiter, 1987] proved that minimal diagnoses can be computed from minimal conflicts. Its main result being that minimal hitting sets of a collection of minimal conflicts yield the minimal diagnoses, where a hitting set of a collection of sets is a set intersecting every set of this collection.

Theorem 1 ([Reiter, 1987], Th. 4.4)**.** *$D \in COMPS$ is a diagnosis for $(COMPS, SD, OBS)$ iff D is a minimal hitting set for the collection of conflict sets for $(COMPS, SD, OBS)$.*

The relation above was established for conflicts as sets of components with no attributes. Each component in a diagnosis hence belongs to one or more conflicts. We say a component *explains*, or equivalently *covers* these conflicts.

Hitting set algorithm

An incremental algorithm to generate all the minimal hitting sets based on a set of conflicts was originally proposed by [Reiter, 1987], then corrected by [Greiner *et al.*, 1989]. This algorithm gives a means to compute diagnoses incrementally, under the permanent fault assumption.

The diagnosis algorithm builds a Hitting-Set tree (HS-tree) in which all the nodes but leaves are labelled by a conflict set. For each component C in the conflict label of node n, an edge labelled C joins n to a successor node. $H(n)$ is defined as the set of edge labels on the path from n to the root node. The HS-tree is built by considering every conflict in arbitrary order. Every new conflict is compared to every leaf of the HS-tree, and some new leaves are built if necessary. The resulting HS-tree is pruned for redundant or subsumed leaves before the next conflict is considered. Pruned leaves are said to be closed. At the end of the diagnosis procedure, the minimal hitting sets, and hence the minimal diagnoses that explain the system's misbehaviors, are given by the set of edge labels $H(l)$ associated to the open leaves l of the HS-tree.

A water transport system example

We consider the simple example of a two reservoirs system, pictured on Figure 1. A system consists in continuously supplying water to two consume areas (o_1 and o_2 are the corresponding flows) from two cascaded geographically distant reservoirs $C1$ and $C2$ (y_1 and y_2 are the water levels in the respective reservoirs). The water transport between reservoirs is modeled as an open flow channel. A hand-switch regulates the water pressure by channeling it either through a pipe $C3$, or through a pump $C4$. Assume that o_1 and o_2 can be measured. In the full application, the system is modeled by a set of discrete time equations that need not to be presented here. So $COMPS = \{C1, C2, C3, C4\}$ and $OBS = \{o_1, o_2\}$.

Example 1 Assume o_1 is measured and found discrepant while water is channeled through $C3$. This produces conflict $s_1 = \{C1, C2, C3\}$. Similarly, assume o_2 is measured and found discrepant while water is pumped through $C4$. This discrepancy leads to conflict $s_2 = \{C1, C2, C4\}$. This problem has three minimal diagnoses given by the hitting sets for $\{C1, C2, C3\}$ and $\{C1, C2, C4\}$: $D_1 = \{C1\}$, $D_2 = \{C2\}$, $D_3 = \{C3, C4\}$.

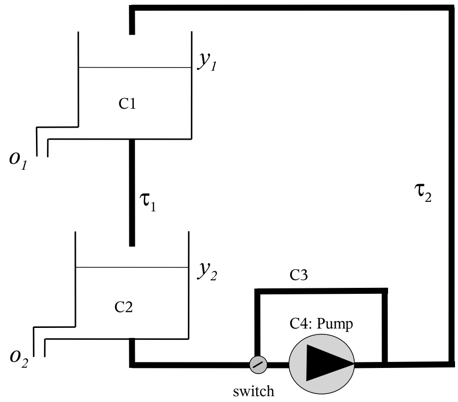


Figure 1: Example 1 & 2: a water transport system (Example 1: with no delays; Example 2: with delays).

2.2 Continuous generalization of conflicts and diagnoses

In many real-world applications, the conflicts contain more information than the components. Thus an attribute to a component C in a conflict can be any additional piece of information, either temporal, such as *C has been faulty for at most four units of time*, or functional, such as *C is faulty with a certain parameter in value range* $[a, b]$. Taking such information into account necessitates an extension to the existing framework.

Continuous extent

To begin with, [Dressler and Freitag, 1994] define the temporal extent $TE(\alpha)$ of a proposition α as the set $\{t \mid \alpha \text{ holds at } t\}$. Following the literature, let us denote the fact that $C \in COMPS$ is faulty by the predicate $AB(C)$. Then $TE(AB(C))$ is the set of time points at which C is faulty. Unfortunately this approach is limited to points in a continuous space. In application, attributes come in more general forms. Consider a continuous space $\mathcal{X}$ of finite dimension. We adopt a more general approach to modeling continuous information in the model-based diagnosis framework. In this novel formulation, a proposition attribute can be any point, interval (bounded or unbounded) or set of intervals (convex or non convex), within $\mathcal{X}$. We name this extension the *continuous extent* $CE(\alpha)$ of a proposition α.

Definition 1 (Continuous extent)**.** *The continuous extent of a proposition α, $CE(\alpha)$, denotes the continuous region $\{\Omega \in \mathcal{X} \mid \alpha \text{ holds in } \Omega\}$.*

We use the following notation: given an interval $I \in \mathcal{X}$, C_I is a short way to represent $CE(AB(C)) \subseteq I$.

Example 2 Consider again the exemple on Figure 1, and consider the additional temporal information. τ_1 between the two reservoirs and τ_2 between the pump and reservoir 1 are the transport time delays, with $\tau_1 \geq 0, \tau_2 \geq 0$. Water traverses a reservoir in 1 unit of time. In this example, the information on the failure time of a component C, relatively to the current time, is a continuous extent of the proposition $AB(C)$.

Conflicts and diagnoses with continuous extents

In context, each component Ci of $COMPS$ operates over a continuous line x_i, where i is the component index. This line is a continuous extent line that supports the component failure time, or any fault related parameter or state variable values. In case where it is bounded, it is referred as the continuous extent domain of Ci. Thus $\mathcal{X} = \bigotimes_{i=1}^{M} x_i$ where M is the number of components in $COMPS$. Here we assume there is a single continuous extent line x_i per component C_i. This assumption allows to simplify notations, but it has no incidence on the generality of the formalism and the computational method proposed in this paper.

We assume component Ci in a conflict s has a known unidimensional failure interval $I_i^s \subseteq x_i$. Following definition 1, $Ci_{I_i^s}$ denotes a failure of Ci within interval I_i^s. To simplify notations Ci_j where j is a real or an integer denotes $Ci_{[j,+\infty[}$. A conflict of cardinality k defines a continuous region $\bigotimes_{i=1}^{k} I_i^s$ that is a hypercube of $\Re^k$. Given a collection S of conflicts, a diagnosis of cardinality k is a tuple (D, X) where D and X are the discrete elements and associated continuous extents, respectively. Thus a minimal diagnosis $D = \{C_1, \cdots, C_k\}$ with continuous extent defines a minimal continuous region $X = \bigotimes_{i=1}^{k} I_i$ that is a hypercube of $\Re^k$. This region is equivalently referred to as the minimal continuous extent of diagnosis D, or a minimal continuous diagnosis.

Example 2 (continued). Assume o_1 is measured and found discrepant while water is channeled through $C3$. By tracing back the system's delays, this produces conflict $s_1 = \{C1_1, C2_{\tau_2+1}, C3_{\tau_2+1}\}$. This is because o_1 discrepant implies that $C1$ was faulty at least a unit of time ago, and/or $C2$ and/or $C3$ at least $\tau_2 + 1$ units of time ago. Similarly, assume o_2 is measured and found discrepant while water is pumped through $C4$. This discrepancy leads to conflict $s_2 = \{C1_{\tau_1+1}, C2_1, C4_{\tau_1+\tau_2+2}\}$.
Then, $(D_1, X_1) = (C1, x_1 \in [\tau_1 + 1, +\infty[)$ and $(D_2, X_2) = (C2, x_2 \in [\tau_2 + 1, +\infty[)$ are two diagnoses. (D_1, X_1) assumes $C1$ has been faulty long enough, i.e. for at least $\tau + 1$ units of time, so it can explain both conflicts s_1 and s_2. $(D_3, X_3) = (\{C1, C2\}, x_1 \in [1, \tau_1 + 1[\cup x_2 \in [1, \tau_2 + 1[)$ assumes that (i) $C1$ has not been faulty for long enough so that it can explain s_2; (ii) $C2$ has not been faulty for long enough so that it can explain s_1. Overall, there are three more diagnoses that in Example 1:

$$(D_4, X_4) = (\{C1, C4\}, x_1 \in [1, \tau_1 + 1[\cup x_4 \in [\tau_1 + \tau_2 + 2, +\infty[)$$
$$(D_5, X_5) = (\{C3, C2\}, x_2 \in [1, \tau_2 + 1[\cup x_3 \in [\tau_2 + 1, +\infty[)$$
$$(D_6, X_6) = (\{C3, C4\}, x_3 \in [\tau_2 + 1, +\infty[\cup x_4 \in [\tau_1 + \tau_2 + 2, +\infty[).$$

Proposition 1. *Given a collection S of conflicts, a diagnosis (D, X) is such that*

$$X \subseteq \bigotimes_{i=1}^{k} [\bigcap_{s \in S_i^*} I_i^s] \qquad (1)$$

where $S_i^* \subseteq S$ is the subset of conflicts that are explained *by Ci.*

Proof. Consider a collection S of conflicts. Then for every component $Ci \in D$, there are three non-exclusive cases:

1. It exists at least s st. $I_i^{s'} \subseteq I_i^s$, for all $s' \in S$. In this case, Ci explains all conflicts in S with $I_i = \bigcap_{s \in S_i^*} I_i^s$ and $S_i^* = S$.
2. $\exists s \in S, s' \in S$ st. $I_i^s \neq I_i^{s'}$ and $I_i^s \cap I_i^{s'} = I$ with $I \neq \emptyset$. In this case
$$C_i \text{ explains } \begin{cases} s \text{ and } s' & \text{for } I_i = I \text{ and } S_i^* = \{s, s'\}, \\ s & \text{for } I_i = I_i^s - I \text{ and } S_i^* = \{s\}, \\ s' & \text{for } I_i = I_i^{s'} - I \text{ and } S_i^* = \{s'\}. \end{cases}$$
3. the I_i^s are all disjoints for all $s \in S$. In this case, Ci explains conflict s iff $I_i = I_i^s$ and $S_i^* = \{s\}$.

It follows that minimal diagnoses are either such that $I_i = \bigcap_{s \in S_i^*} I_i^s$, in case 1 and 3, and $I_i \subset \bigcap_{s \in S_i^*} I_i^s$ in cases 2. Since $X = \bigotimes_{i=1}^{k} I_i$, it comes (1). □

The relation (1) is a consequence of having conflicts with overlapping continuous extents. This proposition conditions diagnoses upon regions that are smaller than the continuous extent in each conflict.

Example 2 (continued). Consider diagnosis $(D_3, X_3) = (\{C1, C2\}, x_1 \in [1, \tau_1 + 1[\cup x_2 \in [1, \tau_2 + 1[)$ for the conflicts $s_1 = \{C1_1, C2_{\tau_2+1}, C3_{\tau_2+1}\}$ and $s_2 = \{C1_{\tau_1+1}, C2_1, C4_{\tau_1+\tau_2+2}\}$. Proposition 1 applies: $C1$ explains s_1, and $C2$ explains s_2, with $x_1 \in [1, \tau_1+1[\subset [1, +\infty[$ and $x_2 \in [1, \tau_2 + 1[\subset [1, +\infty[$.

Reiter's main result applies to the extended representation.

Theorem 2. *(D, X) is a diagnosis for $(COMPS, SD, OBS)$ iff D and X are minimal hitting sets for the collection of conflict sets with continuous extent for $(COMPS, SD, OBS)$.*

Example 2 (continued). There are six diagnoses that are the hitting sets for the conflicts with continuous extents $s_1 = \{C1_1, C2_{\tau_2+1}, C3_{\tau_2+1}\}$ and $s_2 = \{C1_{\tau_1+1}, C2_1, C4_{\tau_1+\tau_2+2}\}$:

$$(D_1, X_1) = (C1, x_1 \in [\tau_1 + 1, +\infty[)$$
$$(D_2, X_2) = (C2, x_2 \in [\tau_2 + 1, +\infty[)$$
$$(D_3, X_3) = (\{C1, C2\}, x_1 \in [1, \tau_1 + 1[\cup x_2 \in [1, \tau_2 + 1[)$$
$$(D_4, X_4) = (\{C1, C4\}, x_1 \in [1, \tau_1 + 1[\cup x_4 \in [\tau_1 + \tau_2 + 2, +\infty[)$$
$$(D_5, X_5) = (\{C3, C2\}, x_2 \in [1, \tau_2 + 1[\cup x_3 \in [\tau_2 + 1, +\infty[)$$
$$(D_6, X_6) = (\{C3, C4\}, x_3 \in [\tau_2 + 1, +\infty[\cup x_4 \in [\tau_1 + \tau_2 + 2, +\infty[).$$

Here diagnosis $\{C1\}$ explains both conflicts iff $x_1 \in [\tau_1 + 1, +\infty[$. Otherwise, $\{C1\}$ cannot explain s_2. Therefore an additional component is needed for a diagnosis to explain the conflicts when $x_1 \in [0, \tau_1 + 1[$. Also, the attentive reader notices that $\{C1\}$, $\{C2\}$, $\{C3, C4\}$, the diagnoses previously obtained on Example 1 with no continuous extent are present in the solution above. However, three new diagnoses have been found: $\{C1, C2\}$, $\{C1, C4\}$, $\{C3, C2\}$. Obviously, these diagnoses are the consequence of the presence of additional information in the form of continuous extents. To each minimal diagnosis corresponds a minimal region of $\mathcal{X}$.

By characterizing diagnoses with continuous extents, Theorem 2 provides the basis for the computation of diagnoses. Therefore, finding the minimal diagnoses and associated minimal continuous regions is the aim of the continuous extension to the HS algorithm that is presented in the next section.

3 Generating Diagnoses from Conflicts with Continuous Elements

3.1 Solution Approach

The original hitting set algorithm considers conflict sets with discrete elements only. It looks for singletons in each conflict set. The algorithm cannot condition the diagnoses upon the different continuous failure points of a component. As briefly shown in the previous section, doing this significantly enlarges the number of diagnoses. It follows that the difficulty we address in this paper is the potentially huge size of the space of diagnoses over continuous regions. The reason for this size is the existence of continuous variables. The hitting set algorithm is exponential in the number of conflict elements so the number of potential diagnoses is staggering.

Underlying the diagnoses are the conflicts, each being explained by the failure of a component in certain regions of its continuous line. It follows that the dimension of the continuous space is the total number of different components in the set of conflicts. In general we assume the continuous space dimension to be equal to the number of components in the system. The challenge is thus to apply the hitting set algorithm to this continuous state-space. Our solution to address this issue is to search for minimal diagnoses in an *aggregate space of diagnoses*. This space is represented by a directed acyclic graph (DAG) in which there is a node for each potential diagnosis discrete element. In other words, each node of our DAG represents a continuous region in which the discrete diagnosis element is the same. Given such a partition of the continuous space, the CHS uses a DAG structure to build the minimal continuous diagnoses that are the hitting sets of the collection of conflicts with continuous extents.

To take advantage of the aggregated representation of the space, the standard HS algorithm must be modified in important ways. In particular, there is no longer a unique correspondence between a node and a diagnosis (D, X). Each node of the DAG now supports the continuous diagnoses in X that have an identical discrete element D. A consequence is that different conflicts can be explained in different regions of a node's continuous region. To address this problem and find the hitting sets, the nodes of the DAG receive a set of functions of the continuous space $\mathcal{X}$ that allow to map different conflicts to different continuous regions. This reflects in the more complex data structures required by the CHS.

In summary, a simple way of understanding the CHS algorithm is as a variant of the HS algorithm where for every conflict, candidate diagnoses with identical discrete elements are expanded in unison. The main difference with the HS is threefold:

- In the standard HS-tree, a single diagnosis is associated with each node. In the CHS, multiple diagnoses are associated with a single node.
- The CHS produces a DAG instead of a tree.
- Nodes are often simultaneously a leaf and a node in the interior of the DAG. This is a consequence of having different conflicts explained in different continuous regions of the same node. In particular, this happens when a part of the aggregated diagnoses do explain all conflicts, while another part does not.

3.2 Data Structures

The main data structure represents a node n. Given a set of conflicts S, it contains:

- A diagnosis $H(n)$ that is a set of k_n edge labels, i.e. components.
- A region X_n of continuous diagnosis elements. It represents the continuous lines of the components in $H(n)$, $X_n \subseteq \Re^{k_n}$.
- $Open_n(.) \rightarrow \{0, 1\}$: the *Open* function. For each $x \in X_n$, $Open_n(x)$ indicates whether (n, x) explains all conflicts in S. The open region of n is noted $\Omega_n = \{x \in X_n | Open_n(x) = 1\}$. A diagnosis is either opened or closed. Note that we don't refer to open or closed nodes; instead we refer to diagnoses associated with nodes as being open or closed.
- $\delta_n(., .)$ the *explanation function*. For $x \in X_n$ and $s \in S$, $\delta_n(s, x)$ indicates whether s is explained by (n, x). Formally,

$$\delta_n(s, x) = \begin{cases} 1 & \text{if } \exists Ci_{I_i^s} \text{ st. } Ci \in s \cap H(n) \text{ with } x \in I_i^s, \\ -1 & \text{otherwise.} \end{cases} \tag{2}$$

3.3 The CHS algorithm

Algorithm 1 presents the main procedure.

Expansion (lines 9 to 12):

For a node n and a conflict s:

$$A_n(s, x) = \begin{cases} \{C \in s | C \notin H(n)\} & \text{if } \delta_n(s, x) = -1 \\ \emptyset & \text{otherwise} \end{cases} \tag{3}$$

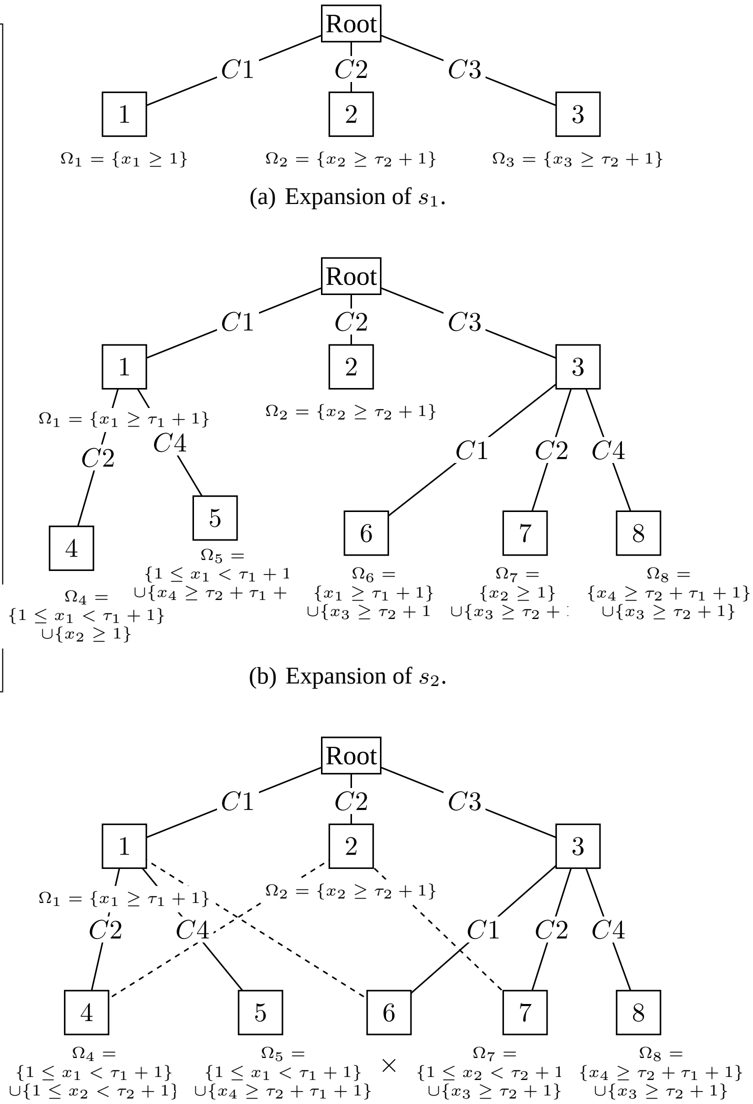
is the set of discrete conflict elements that can *expand* (n, x). At each iteration, CHS expands a diagnosis (n, x) if it doesn't explain the conflict s. An important distinction between HS and CHS is that in the latter, nodes are often partially expanded. This means not all conflicts are explained by some diagnoses (n, x) of node n. The catch is that only those (n, x) that do not explain all conflicts are expanded, and closed after expansion.

Computing the explanation functions (lines 10 & 4):

Each newly expanded (n, x) must be updated. This consists in recomputing its explanation function (Eqn (2)).

1: Root node such that $H(Root) = \{\}$ and $\Omega_{Root} = \Re^M$, for M components in SD.
2: **for all** conflict sets $s \in S$ **do**
3:   **for all** (n, x) such that $Open_n(x) = 1$ **do**
4:     **if** $\delta_n(s, x) = 1$ **then**
5:       For all $C \in H(n) \cap s$, add the pair (n, x) to *oldleaves*$[C]$.
6:     **else**
7:       **for all** $C_{I^s} \in s$ **do**
8:         **if** $A_n(s, x)$ contains C **then**
9:           Expand (n, x) by adding an edge labelled with C and successor aggregated nodes $\big(n', x' = (x, y)\big)$ with $y \in I^s$.
10:           Compute $\delta_{n'}(s, x')$, open / close (n', x') accordingly.
11:           Add the pair (n', x') to *newleaves*$[C]$.
12:         Close the expanded (n, x).
13:   **for all** C in s **do**
14:     **for all** leaf (n, x) of *newleaves*$[C]$ **do**
15:       **if** $H(n)$ contains $H(n')$ and Ω_n contains $\Omega_{n'}$ for some (n', x') in *oldleaves*$[C]$ **then**
16:         Close (n, x).

Algorithm 1: CHS algorithm.



(a) Expansion of s_1.

(b) Expansion of s_2.

(c) Pruning after expansion of s_2.

Figure 2: Expansion and pruning.

Opening & closing of continuous regions (line 10):
The algorithm proceeds by leaving open the regions of the continuous space that explain all conflicts, and by closing the others. Similarly to the original HS, the CHS has an expansion phase and a pruning phase. In the expansion phase, (n, x) is closed if it has been expanded, or if $\exists s$ st. $\delta_n(s, x) = -1$ and $A_n(s, x) = \{\}$. In the pruning phase, (n, x) is closed if it is subsumed by some other node (n', x') such that $H(n') \subseteq H(n)$ and $\Omega_{n'} \subseteq \Omega_n$. For every new conflict s and every element C of the conflict, the algorithm builds two lists, *newleaves[C]* and *oldleaves[C]*, which are then compared. Closed regions of a given node cannot be reopened. This is easily seen since closed regions contain points that do not explain all conflicts. Therefore these regions are expanded into new nodes. The (n, x) that remain opened after all conflicts in S have been processed are the minimal diagnoses.

Example:
Consider the two conflicts of example 2.1. Figure 2(a) pictures the CHS structure after the expansion of s_1. Expansion of s_2 leads to the closing of the subregion $1 \leq x_1 < \tau_1 + 1$ of node 1, and closes node 3, see Figure 2(b). Node 2 is unchanged since after step 4, $\delta_2(s, x_2) = 1$ for all open $x_2 \geq \tau_2 + 1$, leaving $A_2(.)$ empty. The pruning phase closes nodes and regions. A node n is closed whenever for all x, $\Omega_n(x) = \emptyset$ for all $x \in X_n$. Node 6 is closed as it is subsumed by node 1. Similarly, node 2 subsumes some continuous regions of nodes 4 and 7, that are thus closed. Node inclusions are represented with dashed edges on Figure 2(c).

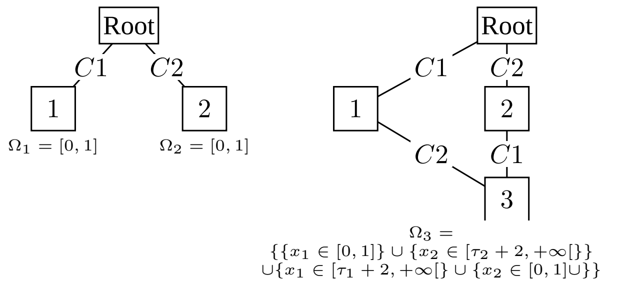


Figure 3: CHS produces a DAG. Left: expansion of $s_1 = \{C1_{[0,1]}, C2_{[0,1]}\}$. Right: expansion of $s_2 = \{C1_{\tau_1+2}, C2_{\tau_2+2}\}$, $\tau_1 \geq 0, \tau_2 \geq 0$, & pruning.

DAG:
The CHS produces a DAG. There exist multiple paths from the Root node to some other nodes. Note that the DAG structure allows disjoint diagnosis regions to be aggregated in the same node (see Figure 3).

3.4 Handling Continuous Variables

Computationally, one challenging aspect of the CHS is the handling of continuous variables. As previously mentioned, for n, and $H(n)$ of cardinality k_n, $X_n \subseteq \Re^{k_n}$. In algorithm 1, the expansion phase replicates the continuous state-space of a father node n into a child node n', such that $X_n \subset X_{n'} \subseteq \Re^{k_n+1}$. In practice it is possible to maintain a single multidimensional space in $\Re^M$ where M is the total number of components in *SD*. In this space, each conflict is a hypercube of dimension $\leq M$. Step 2 of the CHS can be implemented as an intersection of all conflict hypercubes. This results into a partitioned hypercube of dimension M. Remaining operations translate into a labelling/unlabelling of the cube regions with the diagnoses of open nodes. In implementation we use bsp-trees and the intersection operator in [Friedman *et al.*, 1977].

3.5 Properties

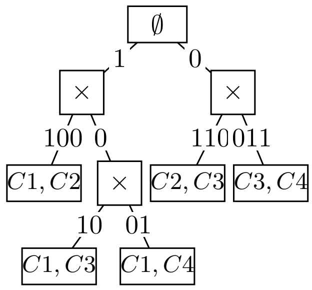
Theorem 3 (Soundness of CHS)**.** *Let S be a set of conflict sets, and T a CHS-DAG obtained by using the CHS with node closing and pruning. For any open node n of T, $(H(n), \Omega_n)$ is a minimal hitting set for S.*

Proof. Steps 4 and 8 ensure that any open (n, x) is a hitting set: it hits every conflict. It remains to show that (n, x) is minimal. Assume that (n, x) is not minimal. Then it exists an open node (n', x') that is such that $H(n') \subseteq H(n)$ and $\Omega_{n'} \subseteq \Omega_n$ and that is not in T. The CHS builds nodes from sets to supersets. Therefore n' must have been generated before n and must be in T. So there is a n' such that $H(n') \subseteq H(n)$. Thus either n or some other node n'' was expanded from n'. This means there exist some $x' \in X'$ such that they do not explain all conflicts in S. But in this case, the (n', x') are closed, either expanded or subsumed. Thus the x' are not in $\Omega_{n'}$. So by contradiction of the assumption, (n, x) is minimal. □

Theorem 4 (Completeness of CHS)**.** *Let S be a set of conflict sets, and T a CHS-DAG obtained by using the CHS with node closing and pruning. For any minimal hitting set (H^*, X^*), there exists an open node n of T such that $(H(n), \Omega_n) = (H^*, X^*)$.*

Proof. Assume (H^*, X^*) minimal hitting set of size k. Then there must be k components over S conflicts such that for every conflict $s \in S$, $s \cap H^* \neq \emptyset$. By construction of the DAG, for each conflict s CHS updates open nodes whose intersection with s is not empty, and expands all other open nodes (n, x). So given the consideration above, there exists a path from the Root node to (H^*, X^*). This path goes through successively created nodes. However, all these nodes must be closed either: i/after being expanded into other nodes; ii/if subsumed by some other nodes, which is impossible if (H^*, X^*) is minimal. So (H^*, X^*) marks the end of the path and is opened. In case $k = 0$, the Root node is the returned solution. □

Alike the HS, CHS is incremental and takes conflicts in any order. Searching for all hitting sets of a given set is NP-complete, and the worst case performance of the standard HS is in the order of 2^M. In fact, the observed performances are usually well under this theoretical bound but more realistic bounds of the HS performances are difficult to obtain. For the CHS, three cases can be distinguished, where in each conflict: i) each component has a single failure point; ii) each component has a single failure interval; iii) each component has disjoint failure intervals. The complexity of mixtures of these situations lies within the theoretical bounds for i), ii) and iii).



(a) Leaves (C1,C2),(C1,C4),(C1,C3),(C2,C3),-(C3,C4) of example 1 before pruning.

insertion lookup table		
edge	key	lookup
1	1	continue
1	0	continue
0	0	continue
0	1	stop

(b) Insertion lookup table.

Figure 10: P-tree storage.

Assume M components over a set of K conflicts. The number of occurence of component m over all conflicts is noted $0 \geq f_m \leq K$. In case ii), for m, the maximum number of intervals over all conflicts is $2f_m - 1$. This corresponds to the case where all intervals for m in conflicts do intersect with each others. Each intersected region thus explains a different subset of conflicts, and corresponds to different nodes of the CHS-DAG. Consequently an upper bound to the worst case performances is given by $\sum_{m=1}^{M} [2f_m - 1] + \binom{M}{2} \prod_{m_i, m_j} [2f_{m_i} - 1][2f_{m_j} - 1] + \cdots + \prod_{m=0}^{M} [2f_m - 1]$.

With it, bounds on cases i) and iii) can be easily expressed by considering unbounded intervals, and a fixed number of intervals per component, respectively.

4 Computational Improvements

In this section we propose two computational improvements to the CHS algorithm. One improves on the pruning computational efficiency, and the second yields a controlled approximation.

4.1 Efficient Implementation of the Pruning Loop

A dominant source of complexity of the CHS is the final closing of leaves (lines 13 to 16). The CHS creates new leaves for each conflict. In the worst case, there are md newly created leaves where d is the number of leaves before expansion and $m \leq M$. Thus a naive implementation leads to a procedure that requires $O(d^2 m)$ inclusion checks (step 15). We develop a tree-based data structure that supports faster inclusion checks. The basic idea is to dissociate the inclusion check of the discrete element of diagnoses from the continu-

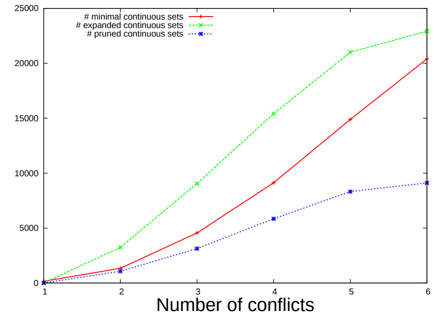


Figure 4: Mean minimal continuous diagnoses wrt. the number of conflicts.

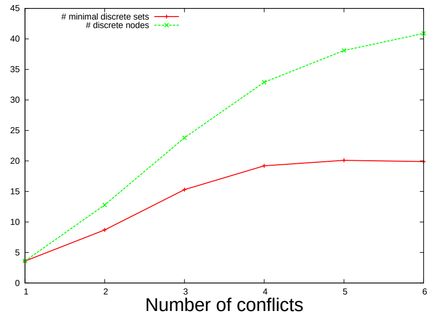


Figure 5: Mean minimal discrete diagnoses wrt. the number of conflicts.

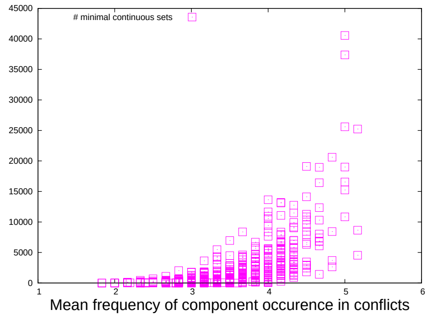


Figure 6: Minimal continuous diagnoses (500 runs).

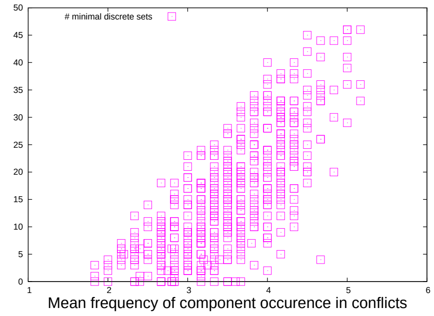


Figure 7: Minimal discrete diagnoses (500 runs).

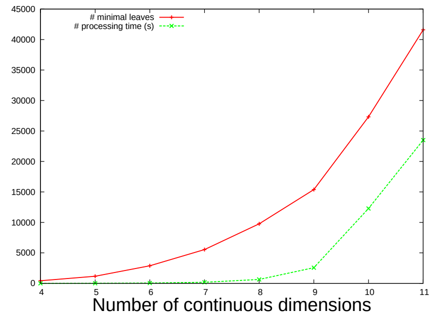


Figure 8: Minimal continuous diagnoses.

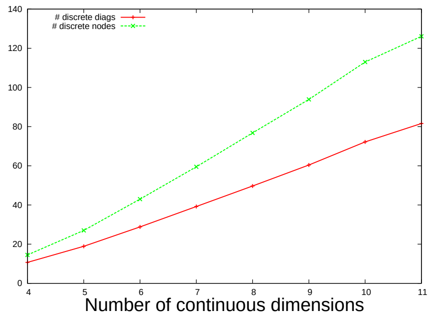


Figure 9: Minimal discrete diagnoses.

ous set inclusion check. By using an adaptation of the PATRICIA tree (P-tree) [Morrison, 1968], the (n, x) in *newleaves* are grouped based on their discrete component n in a P-tree. The newly created leaves of the CHS-DAG are located in the leaves of the P-tree. The P-tree stores node n with an M-bit key where bit i is positive if $Ci \in H(n)$. Figure 10 depicts such a tree for the *newleaves* list of example 1.

Insertion cost is $O(M)$ bit checks for each new leaf. A special operator performs an inclusion lookup for a node n' that returns all n such that $H(n') \subseteq H(n)$. For an M-bit key with k^0 negative bits and inclusion lookup table 10(b), an inclusion lookup costs 2^{k^0} positive bit checks, and $M - k^0$ negative bit checks per explored tree branch, so total cost is $O(2^{k^0}(M - k^0))$ bit checks. Clearly, the advantage is that with this storage structure, the inclusion lookup does not depend on d or the number of conflicts. Thus in practice we observe a good speed-up of the pruning loop that improves the CHS scalability. See the result section for empirical details. The inclusion test is performed for each of the *oldleaves* elements. The continuous inclusion check is performed on those new leaves that are returned by each inclusion lookup.

4.2 Generating Relevant Diagnoses

In Artificial Intelligence it is important to study the generation of approximated results. Here the idea is developed that some nodes of the CHS DAG are more important than others. Suppose that each component C has a probability distribution $p\big(AB(C)|x\big)$ of failing over x. The probability of a node (n, x) is:

$$p_n(x) = \prod_{Ci \in H(n)} p\big(AB(Ci)|x\big) \quad (4)$$

and the probability of n is:

$$p_n = \int_{\Omega_n} p_n(x)dx. \quad (5)$$

The CHS algorithm can be easily adapted to the computation of the most relevant minimal diagnoses. Given a number ϵ between 0 and 1, the nodes (n, Ω_n) with $p_n < \epsilon$ are closed.

5 Experimental Evaluation

The CHS was implemented and tested extensively through simulation experiments. Overall, it yields fast results for spaces under 10 dimensions, but doesn't scale favorably well beyond. The main results are drawn from a set of 500 runs of the CHS for $M = K = 6$ components and conflicts. The simulation settings were as follows. Each conflict has a random size. Each component in a conflict comes with a random interval that is generated by picking up two integers between 0 and 100.

This section reports on the reactions of the CHS. The continuous diagnoses are the open continuous leaves of the CHS-DAG and their number is the total number of minimal diagnoses. The discrete diagnoses are the open nodes of the CHS-DAG. Both numbers theoretically grow exponentially with the total number of conflict elements. This is visible on Figure 4 despite the fact that our experiments were limited to small numbers of components. In consequence the CHS has expanded many of the discrete diagnoses after just a few conflicts (Figure 5). The DAG structure in the aggregate space of diagnoses allows the minimal continuous diagnoses to continue to grow after all discrete diagnoses have been expanded (Figure 4).

The complexity analysis has shown how the number of occurence of a component in conflicts plays a crucial role. This is clearly confirmed on Figure 6. The explosion of the number of minimal continuous diagnoses is a direct consequence of the NP-complete nature of the problem. Figure 7 shows the minimal discrete diagnoses are distributed differently. This is due to the DAG structure: given a mean integer f of mean occurences over M components, this number is always smaller than $\sum_{i=1}^{f} \binom{M}{i}$. That is, the number of minimal discrete diagnoses is at most all combinations of f and fewer components.

Based on a second set of experiments we aimed to elucidate the scaling properties of the approach wrt. the continuous dimensions. These experiments are runs with M ranging from 1 to 11, $K = 4$, and conflict random intervals in $[0, 10]$. The results are graphically depicted on figures 8 and 9. The exponential response of the number of minimal continuous diagnoses appears clearly.

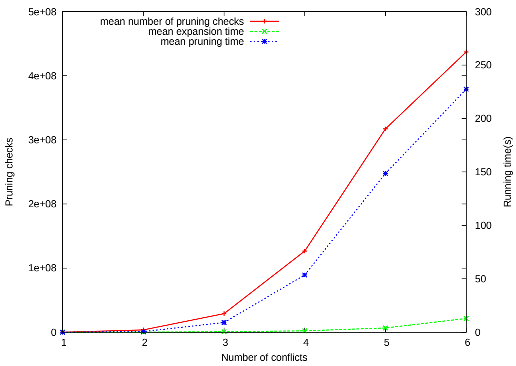


Figure 11: Expansion vs. pruning.

In practice it was not possible to run the CHS in reasonable time on problems with more than 10 or so continuous dimensions. The limitation stems mainly from the pruning phase that dominates the computational effort (Figure 11). In worst cases however, the pruning loop continues to require up to several billions inclusion checks of continuous sets.

6 Conclusion

We have presented the CHS algorithm, a solution to finding the minimal hitting sets of a collection of finite sets whose elements have continuous extents. The algorithm uses an DAG representation in an aggregate space of diagnoses. CHS is based on the same dual mechanism as the classical hitting set algorithms: it has an expansion phase and a pruning phase. To our knowledge CHS is the first computational method to produce minimal diagnoses conditioned over continuous regions. In practice however, CHS exhibits an unfair behavior: it expands high numbers of potential diagnoses in little time and spends most of its time pruning out a large fraction of them. It is an open problem how to better tackle this computational cost.

References

[Dressler and Freitag, 1994] O. Dressler and H. Freitag. Prediction sharing across time and contexts. In *AAAI94*, Seattle WA, USA, 1994.

[Friedman *et al.*, 1977] J.H. Friedman, J.L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Mathematical Software*, 3(3):209–226, 1977.

[Greiner *et al.*, 1989] R. Greiner, B.A. Smith, and W. Wilkerson. A correction to the algorithm in reiter's theory of diagnosis. *Artificial Intelligence*, 41(1):79–88, 1989.

[Haenni, 1997] R. Haenni. Generating diagnoses from conflict sets. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 1997.

[Lin and Jian, 2003] L. Lin and Y. Jian. The computation of hitting sets: review and new algorithms. *Information Processing Letters*, 186:177–184, 2003.

[Morrison, 1968] D.R. Morrison. Practical algorithm to retrieve information coded in alphanumeric. *Journal of the ACM*, 15(4):514–534, 1968.

[Reiter, 1987] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[Travé-Massuyès and Calderon-Espinoza, 2007] L. Travé-Massuyès and G. Calderon-Espinoza. Timed fault diagnosis. In *European Control Conference (ECC-07)*, 2007.

[Wotawa, 2001] F. Wotawa. A variant of reiter's hitting-set algorithm. *Information Processing Letters*, 79:45–51, 2001.