Generating Diagnoses from Conflict Sets with Continuous Attributes

Emmanuel Benazera and **Louise Travé-Massuyés**[1]

Abstract. Many techniques in model-based diagnosis and other research fields find the hitting sets of a group of sets. Existing techniques apply to sets of finite elements only. This paper addresses the computation of the hitting sets of a group of sets whose elements are convex or non-convex, bounded or unbounded continuous regions. We assume the conflict sets are known and we present a novel procedure, the Continuous Hitting Set algorithm (CHS) for transforming conflict sets of continuous elements 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 [5]. 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 [2] or HS-tree [7] but other representations exist [3, 4]. 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 [6].

In this paper, we address step (ii) and assume all minimal conflicts are given. Each conflict element is a component with 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 conflict sets of continuous attributes is that minimal diagnoses are conditioned upon the component 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 and Solution Approach

2.1 Problem Formulation

We consider a tuple $(COMPS, SD, OBS)$. A component Ci of $COMPS$ operates over a continuous line x_i, where i is the component index, and that represents its failure time, or any other parameter or variable value (bounded or unbounded) domain. A conflict set, or conflict for short, is a set of components which cannot behave normally altogether according to the observations. We assume component Ci in a conflict s has a known unidimensional failure interval I_i^s. It is noted $Ci_{I_i^s}$. To simplify notations Ci_j where j is an integer denotes $Ci_{[j,+\infty]}$. A conflict of cardinality k is noted $s = \{C1_{I_1^s}, \cdots, Ck_{I_k^s}\}$. It defines a continuous region $\bigotimes_{i=1}^k I_i^s$ that is a hypercube of $\Re^k$. A minimal conflict is a conflict that does not strictly include any other conflict. Reiter proved that minimal diagnoses can be computed from minimal conflicts. Each component in a potential diagnosis belongs to one or more conflicts. We say a component *explains*, or equivalently *covers* these conflicts. Given a collection S of conflicts a diagnosis of cardinality k is a tuple (D, X) where D and X are the discrete and continuous diagnosis respectively. They are such that $D \in COMPS$ with $D = \{C1, \cdots, Ck\}$ and $X \subseteq \bigotimes_{i=1}^k [\bigcap_{s \in S_i^*} I_i^s]$, where $S_i^* \subseteq S$ is the subset of conflicts that are *explained* by Ci. The inclusion is a consequence of having conflicts with overlapping continuous regions. This conditions diagnoses upon regions that are smaller than the failure interval in each conflict.

[1] LAAS-CNRS, Université de Toulouse, 7 av. du Colonel Roche, 31077 Toulouse Cedex 4, France. email: ebenazer,louise@laas.fr

Example 1. *Consider two conflicts* $s_1 = \{C1_1, C2_{\tau_2+1}, C3_1\}$, $s_2 = \{C1_{\tau_1+1}, C2_1, C4_1\}$, $\tau_1 \geq 0, \tau_2 \geq 0$. *The six diagnoses are*

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

with $x_1 \in [1, \tau_1 + 1[\subset [1, +\infty[$, $x_2 \in [1, \tau_2 + 1[\subset [1, +\infty[$.

2.2 Solution Approach

2.2.1 *Hitting set algorithm*

A hitting set of a collection of sets is a set intersecting every set of this collection. Minimal hitting sets of a collection of minimal conflicts yield the minimal diagnoses. An incremental algorithm to generate all the minimal hitting sets based on a set of conflicts was originally proposed by [5], then corrected by [2]. 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 element 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.

2.2.2 *Continuous hitting sets*

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. 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* that is represented by a directed acyclic graph (DAG) in which there is a node for each potential diagnosis. In other words, each node of our DAG represents a continuous region in which the discrete diagnosis element is the same.

3 Generating Diagnoses from Conflicts with Continuous Elements

A simple way of understanding the Continuous Hitting Set (CHS) algorithm is as a variant of the HS algorithm where candidate diagnoses with identical discrete elements are expanded in unison. The main difference with the HS is twofold:

- 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 happens when a part of the aggregated diagnoses do explain all conflicts, while another part does not.

In the standard HS-tree, a single diagnosis is associated with each node. In the CHS, multiple diagnoses are associated with a single node.

3.1 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,

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

3.2 The CHS algorithm

Algorithm 1 presents the main procedure.

3.2.1 *Expansion (lines 9 to 12):*

For a node n and a conflict s:

$$A_n(s, x) = \begin{cases} \{C \in s | C \notin H(n)\} & \text{if } \delta_n(s, x) = -1 \\ \emptyset & \text{otherwise} \end{cases} \tag{2}$$

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.

3.2.2 *Computing the explanation functions (lines 10 & 4):*

Each newly expanded (n, x) must be updated. This consists in recomputing its explanation function (Eqn (1)).

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.

3.2.3 *Opening & closing of continuous regions (line 10):*

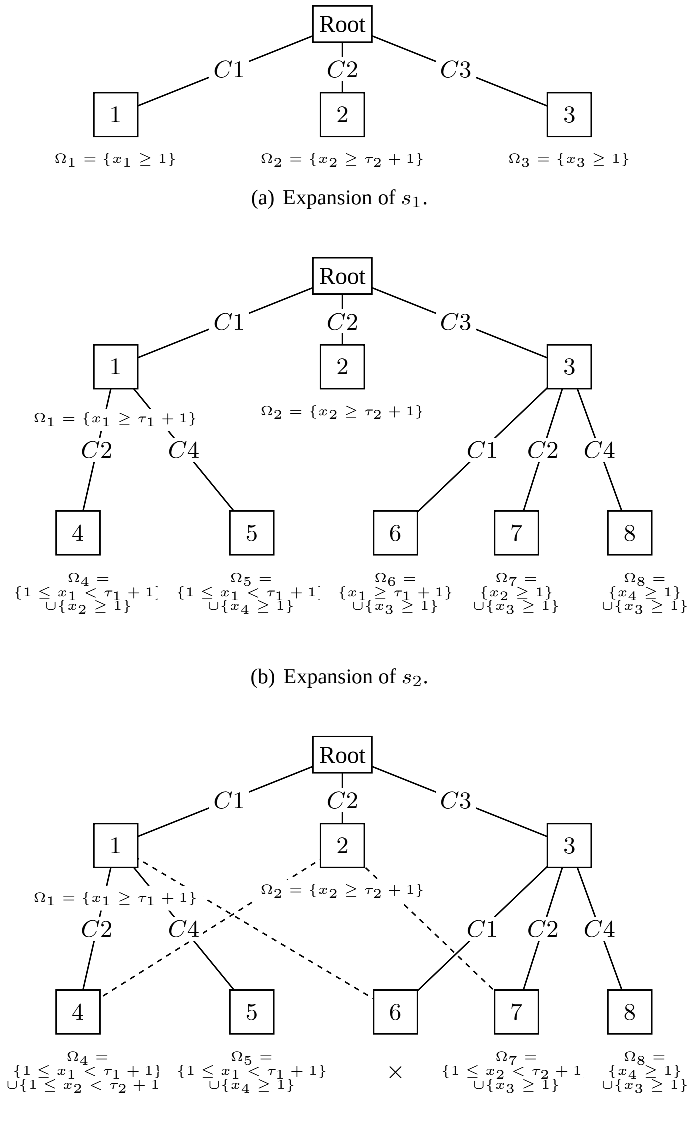
The algorithm proceeds by leaving open the regions of the continuous space that explain all conflicts, and by closing the others. Similarly to the original HS, the CHS has an expansion phase and a pruning phase. In the expansion phase, (n, x) is closed if it has been expanded, or if $\exists s$ st. $\delta_n(s, x) = -1$ 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.

3.2.4 *Example:*

Consider the two conflicts of example 1. Figure 1(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 1(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 1(c).

3.2.5 *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 2).



(a) Expansion of s_1.

(b) Expansion of s_2.

(c) Pruning after expansion of s_2.

Figure 1. Expansion and pruning.

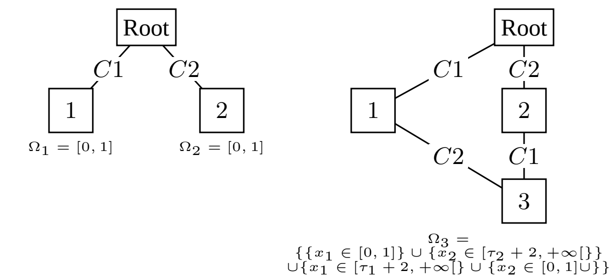


Figure 2. CHS produces a DAG. Left: expansion of $s_1 = \{C1_{[0,1]}, C2_{[0,1]}\}$. Right: expansion of $s_2 = \{C1_{\tau_1+2}, C2_{\tau_2+2}\}$, $\tau_1 \geq 0, \tau_2 \geq 0$, & pruning.

3.3 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 [1].

3.4 Properties

Theorem 1 (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 7 ensure that any open (n, x) is a hitting set. If (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. Moreover if (n', x') is closed, it is either: i/expanded, and thus it exists n'' such that $h(n'') = h(n)$, with n' father of n'' so that $n'' = n$ and n is minimal, which contradicts the hypothesis that n is not minimal; ii/subsumed by some node n'', and thus n is also subsumed by n'', and thus a node that subsumes n has been generated, which contradicts the hypothesis that this node had not been generated. Thus (n', x') is minimal. □

Theorem 2 (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 K conflicts such that for $i = 1, \cdots, k$, $Ci \cap H^* \neq \emptyset$, and $\bigotimes_{i=1}^{k} x_i^* \subseteq X^*$. 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 there exists a path from the Root node to (H^*, X^*). This path goes through successively opened nodes. These nodes are closed either: i/after being expanded into other opened nodes; ii/if subsumed by some other nodes, which is impossible if (H^*, X^*) is minimal. 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).

Assume M components over a set of K conflicts. The number of occurence of component m over all conflicts is noted $0 \leq f_m \leq K$. 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.

3.5 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. Let AB be the *abnormal* predicate such that $AB(C)$ is true whenever C fails. 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:

$$p_n(x) = \prod_{Ci \in H(n)} p\big(AB(Ci)|x\big) \tag{3}$$

and the probability of n is:

$$p_n = \int_{\Omega_n} p_n(x)dx. \tag{4}$$

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.

4 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 designed to validate the theoretical properties of the CHS with no special assumption made on the domain (or SD). As such, they allow a fair examination. The 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 3 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 4). 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 3).

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 5. The explosion of the number of minimal continuous diagnoses is a direct consequence of the NP-complete nature of the problem. Figure 6 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.

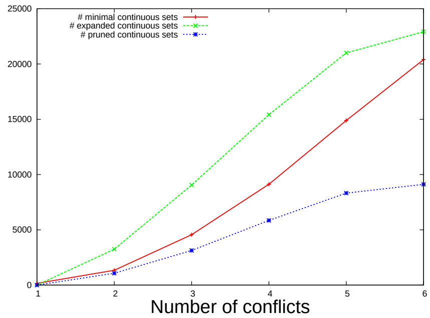


Figure 3. Mean minimal continuous diagnoses wrt. the number of conflicts.

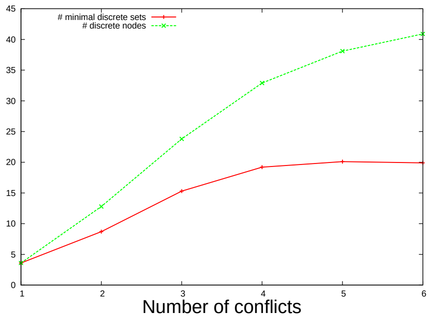


Figure 4. Mean minimal discrete diagnoses wrt. the number of conflicts.

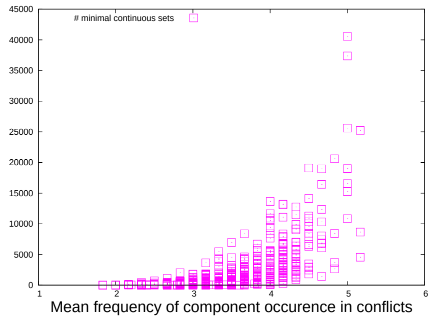


Figure 5. Minimal continuous diagnoses (500 runs).

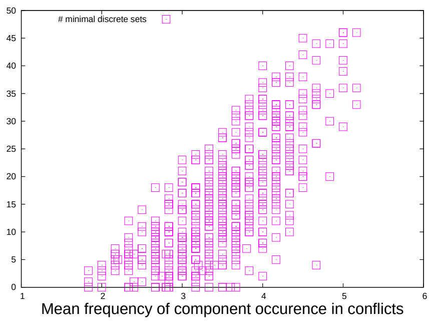


Figure 6. Minimal discrete diagnoses (500 runs).

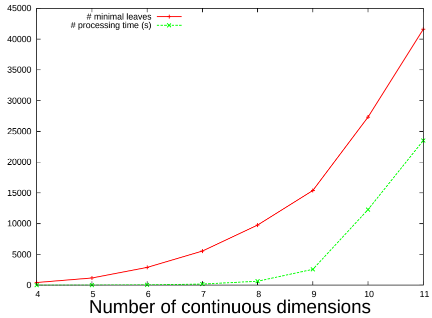


Figure 7. Minimal continuous diagnoses.

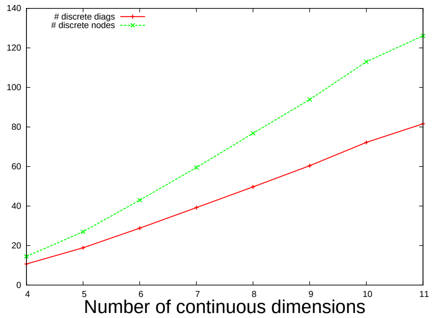


Figure 8. Minimal discrete diagnoses.

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 7 and 8. The exponential response of the number of minimal continuous diagnoses appears clearly.

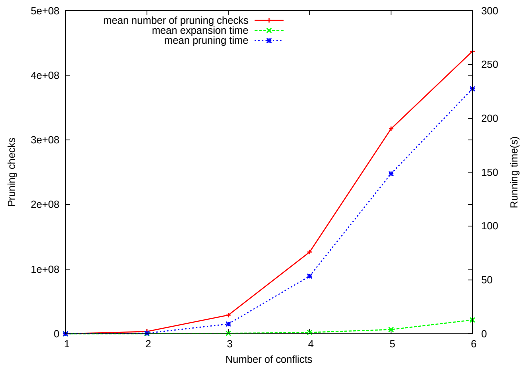


Figure 9. Expansion vs. pruning.

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 9). An addition to the pruning loop allows the inclusion checks of discrete diagnoses to be improved. Space limitation precludes its description here. In worst cases however, the pruning loop continues to require up to several billions inclusion checks of continuous sets.

5 Conclusion

We have presented the CHS algorithm, a solution to finding the minimal hitting sets of a collection of sets with continuous attributes. 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 with continuous attributes. 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

[1] 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).

[2] 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).

[3] R. Haenni, 'Generating diagnoses from conflict sets', in *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, (1997).

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

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

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

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