

# Hierarchical Diagnosis Guided by Observations

Luca Chittaro and Roberto Ranon

Department of Mathematics and Computer Science

University of Udine - 33100 Udine, ITALY

email: {chittaro | ranon}@dimi.uniud.it

## Abstract

We propose a technique to improve the performance of hierarchical model-based diagnosis, based on structural abstraction. Given a hierarchical representation and the set of currently available observations, the technique is able to dynamically derive a tailored hierarchical representation to diagnose the current situation. We implement our strategy as an extension to the well-known Mozetic's approach [Mozetic, 1992], and illustrate the obtained performance improvements. Our approach is more efficient than Mozetic's one when, due to abstraction, fewer observations are available at the coarsest hierarchical levels.

## 1 Introduction

Abstraction has been advocated as one of the main remedies for the computational complexity of model-based diagnosis (MBD). MBD approaches that exploit abstraction (e.g., [Mozetic, 1992]) represent the system at multiple levels of detail, and typically isolate faults one level at a time, starting at the most abstract possible level, and using the results at one level to reduce the cost of diagnosis by removing impossible diagnoses from consideration at more detailed levels.

One limit to the effectiveness of hierarchical diagnosis is the fact that often a single, fixed hierarchical representations is employed, regardless of the currently available observations. In many cases, effectiveness depends on the situation at hand. As we show in the paper, the same hierarchical representation can improve the performance of diagnosis in some cases, but also lead to a performance which is identical to non-hierarchical diagnosis in other cases. On the other hand, building a new hierarchical representation by hand for each diagnostic scenario cannot in general be a viable alternative.

In this paper, we propose a technique that, given a hierarchical representation and a set of currently available observations, is able to derive a hierarchical representation which is particularly suited to diagnose the current situation. Our technique does not build the tailored representation from scratch, but instead *rearranges* the original hierarchical representation, i.e. it exploits only the original abstractions. The rearrangement process is guided by the currently available observations. The technique we present focuses on *structural*

*abstraction* [Hamscher, 1991; Genesereth, 1984]. To show the usefulness of our technique, we formalize it as an extension to Mozetic's approach [Mozetic, 1992], and experimentally evaluate its performance vs. that approach, showing that it improves efficiency when fewer observations are available at the coarsest hierarchical levels due to abstraction.

The paper is structured as follows: Section 2 contains some prerequisites on diagnosis and structural abstraction; Section 3 analyzes Mozetic's approach, while Section 4 presents our hierarchical diagnostic strategy. Section 5 illustrates the experimental evaluation. Finally, Section 6 mentions future work.

## 2 Diagnosis with Structural Abstraction

Following [de Kleer *et al.*, 1992], a diagnosis problem  $D$  in a language  $L$  is a triple  $(SD, OBS, COMPS)$ , where  $SD$  and  $OBS$  are first-order theories in language  $L$ , representing the system description and observations, respectively, and  $COMPS$  is a subset of the object constants of  $L$ , identifying the set of system components. Each component is assigned a set of behavioral modes, each one represented by a distinct predicate  $m$  of  $L$  (and also specified by a formula over the component's ports), that represent either multiple normal operating modes (e.g. a valve can be open or closed) or faulty behavioral modes (e.g. a pipe can be leaking, or be obstructed). Each formula  $[\bigwedge_{c \in COMPS} m_c(c)]$ , where  $m_c$  represents a behavioral mode for component  $c$ , is a *candidate* of  $D$ . A candidate  $\Delta$  is a *diagnosis* of  $D$  if and only if  $SD \cup OBS \cup \Delta$  is consistent. The diagnosis task consists in finding all possible diagnoses, or to find a set of diagnoses (such as the *kernel diagnoses* [de Kleer *et al.*, 1992]) that are a compact representation for all the possible diagnoses.

For clarity purposes, the hydraulic system depicted in Figure 1, called *System A*, will be used as an example throughout the paper. It is composed by a volumetric pump  $pm$ , pipes  $p_1, p_2, p_3, p_4, p_5, p_6$ , and valves  $v_1$  (in closed state) and  $v_2$  (in open state). In Table 1, we define the component modes in *System A* (the formulae that specify the types of components and the connections among them can be easily derived from the schema in Figure 1). In the volumetric pump's behavioral description,  $F_k$ , which depends on the pump's settings, is the amount of flow the normal pump is able to impose. To avoid unnecessary complex examples, we consider only flow observations.

| type         | behavioral modes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>pump</i>  | $(\forall x) \text{pump}(x) \wedge \text{ok}(x) \Rightarrow \text{inflow}(x) = \text{outflow}(x) = F_k$<br>$(\forall x) \text{pump}(x) \wedge \text{leak}(x) \Rightarrow \text{inflow}(x) > \text{outflow}(x)$<br>$(\forall x) \text{pump}(x) \wedge \text{loFlow}(x) \Rightarrow \text{inflow}(x) = \text{outflow}(x) < F_k$<br>$(\forall x) \text{pump}(x) \wedge \text{hiFlow}(x) \Rightarrow \text{inflow}(x) = \text{outflow}(x) > F_k$                                                                                                                      |
| <i>pipe</i>  | $\text{pipe}(x) \wedge \text{normal}(x) \Rightarrow (\text{inflow}(x) = \text{outflow}(x))$<br>$\text{pipe}(x) \wedge \text{leak}(x) \Rightarrow (\text{inflow}(x) > \text{outflow}(x))$                                                                                                                                                                                                                                                                                                                                                                          |
| <i>valve</i> | $\text{valve}(x) \wedge \text{open}(x) \wedge \text{ok}(x) \Rightarrow (\text{inflow}(x) = \text{outflow}(x))$<br>$\text{valve}(x) \wedge \text{closed}(x) \wedge \text{ok}(x) \Rightarrow$<br>$(\text{inflow}(x) = \text{outflow}(x) = 0)$<br>$\text{valve}(x) \wedge \text{leak}(x) \Rightarrow (\text{inflow}(x) > \text{outflow}(x))$<br>$\text{valve}(x) \wedge \text{open}(x) \wedge \text{stuckClosed}(x) \Rightarrow (\text{outflow}(x) = 0)$<br>$\text{valve}(x) \wedge \text{closed}(x) \wedge \text{stuckOpen}(x) \Rightarrow (\text{outflow}(x) > 0)$ |

Table 1: Behavioral modes for the components of System A.

Structural abstraction decomposes a system into a hierarchy of subcomponents (or, vice versa, aggregates components into a hierarchy of supercomponents). We adopt the following terminology and assumptions. The structural abstraction of a diagnosis problem  $D_i = (SD_i, OBS_i, COMPS_i)$ <sup>1</sup> is a (more abstract) problem  $D_{i+1} = (SD_{i+1}, OBS_{i+1}, COMPS_{i+1})$ , obtained by aggregating a subsystem (denoted by the set of components  $AGGR \subseteq COMPS_i$ ) into a single supercomponent  $sc \in COMPS_{i+1}$ .

The more abstract diagnosis problem  $D_{i+1}$  represents the same system as  $D_i$ , but: (i) from a structural point of view, the components belonging to  $AGGR$  have to be replaced by a single component  $sc$ , and, (ii) from a behavioral point of view, a representation of the behavior of  $sc$  has to be introduced. The behavior of  $sc$  is reasonably a simplified representation of the possible combinations of behavioral modes of its subcomponents (e.g., by forgetting and/or collapsing some of the variables of components in  $AGGR$ ). Finally, also the set of observation  $OBS_{i+1}$  could differ from  $OBS_i$ , because observations that are internal to subsystem  $AGGR$  cannot be represented in  $D_{i+1}$ . For example, suppose to aggregate pipe  $p_2$  and open valve  $v_1$  of System A into a component  $se_1$  of type *valve*, in open state. The choice is reasonable since when both  $p_2$  and  $v_1$  are normal, they act as an open valve, allowing fluid to flow. We can thus remove all the formulae which mention  $p_2$  and  $v_1$ , and introduce the theory  $\{\text{valve}(se_1), \text{open}(se_1), \text{outflow}(p_1) =$

<sup>1</sup>In this paper, we adopt the convention of using a greater subscript to indicate a more abstract, less complex theory.

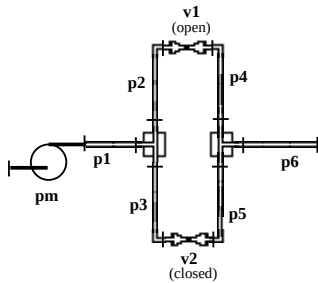


Figure 1: Layout of system A.

$\text{inflow}(se_1) + \text{inflow}(p_3), \text{outflow}(se_1) = \text{inflow}(p_4)\}$ . In this case, the abstraction removes the flow measurement at the connection between  $p_2$  and  $v_1$ , and keeps the other measurements.

We define a *hierarchical representation* of a given diagnostic problem  $(SD_0, OBS_0, COMPS_0)$  as a list of  $m$  diagnostic problems  $\{(SD_i, OBS_i, COMPS_i)\}_m$  with  $i = 0, \dots, m-1$  where for every  $j = 0, \dots, m-2$ ,  $D_{j+1}$  is a structural abstraction of  $D_j$ . A possible 6-levels hierarchical representation of System A is sketched in Table 2. Each row of the table shows a level of the hierarchical representation. For each level, we indicate the level number, the components (together with their type), and the aggregations performed to obtain that level (in the third column of the table,  $c_i + c_j \rightarrow sc$  means that  $c_i$  and  $c_j$  have been aggregated into  $sc$ ).

In order to ensure correctness and completeness when reasoning with abstract system representations, structural abstractions must satisfy some constraints. More specifically, the following property must be satisfied.

**Property 1** (*Relation between diagnoses*). Let  $D_i, D_{i+1}$  be diagnosis problems, such that  $D_{i+1}$  is a structural abstraction of  $D_i$ . Let  $\Delta$  and  $\Gamma$  be candidates of  $D_i$  and  $D_{i+1}$ , respectively, and suppose that  $\Gamma$  is a more abstract representation of  $\Delta$  (according to the aggregation that has been performed in order to obtain  $D_{i+1}$ ). If  $\Gamma$  is not a diagnosis of  $D_{i+1}$ , then  $\Delta$  must not be a diagnosis of  $D_i$ .

If this property is satisfied, one can rule out candidates at the more detailed level by reasoning only at the abstract level, because structural abstraction never mistakenly rules out a diagnosis. On the negative side, diagnosis must consider also the more detailed level in order to ensure correctness and completeness of results. We now show that similar requirements are imposed by other formalizations of structural abstraction proposed in the literature. Thus, the techniques we present are not constrained to a specific formalization. In the following, let  $c_1, \dots, c_k$  be a set of components which is aggregated into the supercomponent  $sc$ , and  $ok(c)$  a formula saying that component  $c$  is behaving normally.

In [Struss, 1992], structural abstraction is defined as follows. Supercomponent  $sc$  is a structural abstraction of  $c_1, \dots, c_k$  if  $ok(c_1) \wedge \dots \wedge ok(c_k) \Rightarrow ok(sc)$ . In this way, whenever  $ok(sc)$  is inconsistent with the observations, then also  $ok(c_1) \wedge \dots \wedge ok(c_k)$  is inconsistent, i.e. by excluding at the abstract level that  $sc$  is normal, we can exclude also that all its subcomponents are normal. This is a special case of Property 1 when we represent only the normal behavior of components in  $SD$ .

[Mozetic, 1992] (using a different formalization of diagnosis) lists a set of *consistency conditions* that must hold between a system representation and its more abstract version. These conditions ensure that Property 1 holds.

[Autio and Reiter, 1998] formalizes structural abstraction for diagnosis problems where only the correct behavior of components is represented. Their approach requires both that  $ok(c_1) \wedge \dots \wedge ok(c_k) \Rightarrow ok(sc)$  (as in Struss' formalization) and that  $ok(sc) \Rightarrow ok(c_1) \wedge \dots \wedge ok(c_k)$  (i.e., if  $sc$  is behaving normally, we can conclude that all its subcomponents are behaving normally). The second assumption can cause

incorrect diagnoses when structural abstraction hides some measurements (e.g. there is a sensor which is “hidden” inside  $sc$ ). Indeed, because of the lacking measurement, one could conclude that the normal behavior of  $sc$  is consistent with the observations, and thus infer that all  $sc$ ’s subcomponents are also behaving normally, which in general is not true.

### 3 Analysis of Mozetic’s approach

We illustrate Mozetic’s diagnostic algorithm in a format that will allow us to easily compare it with our proposal and highlight the refinements we made. The inputs of the algorithm are the hierarchical representation, and a set  $OBS_0$  of observations at the finest level of detail (i.e., level 0). For each level  $i$  of the hierarchical representation,  $Cand_i^{NA}$  is the set of candidates which have no abstraction at upper levels. Hierarchical diagnosis is formalized as follows:

Procedure HIERARCHICAL-DIAGNOSIS  
 ABSTRACT-OBSERVATIONS;  
 TOP-DOWN-DIAGNOSIS.

Procedure ABSTRACT-OBSERVATIONS  
 $l \leftarrow 0$ ;  
 $m \leftarrow$  number of levels of the hierarchical representation;  
 WHILE  $l < m - 1 \wedge OBS_l \neq \emptyset$  DO  
 $OBS_{l+1} \leftarrow \text{Abstract}(OBS_l)$ ;  
 $l \leftarrow l + 1$ ;  
 ENDWHILE;  
 IF  $OBS_l = \emptyset$  THEN  $l \leftarrow l - 1$ .

Procedure TOP-DOWN-DIAGNOSIS  
 $top \leftarrow l$ ;  
 $Cand_l \leftarrow$  all possible candidates at level  $l$ ;  
 WHILE  $l \geq 0$  DO  
 $Diag_l \leftarrow \text{Verify}(SD_l, OBS_l, COMPS_l, Cand_l)$ ;  
 IF  $l < top$  THEN  
 $Diag_l^{NA} \leftarrow \text{Verify}(SD_l, OBS_l, COMPS_l, Cand_l^{NA})$ ;  
 $Diag_l \leftarrow Diag_l \cup Diag_l^{NA}$ ;  
 ENDIF;  
 IF  $l > 0$  THEN  $Cand_{l-1} \leftarrow \text{Detailed}(Diag_l)$ ;  
 $l \leftarrow l - 1$ ;  
 ENDWHILE.

We divided Mozetic’s algorithm into two separate pro-

| level | components                                                                                                     | aggregations                                                     |
|-------|----------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| 5     | $pump(se_7)$                                                                                                   | $se_5 + se_6 \rightarrow se_7$                                   |
| 4     | $pump(se_6), valve(se_5)$                                                                                      | $pm + p_1 \rightarrow se_5$ ,<br>$pa_1 + p_6 \rightarrow se_6$   |
| 3     | $pump(pm), pipe(p_1), pipe(p_6),$<br>$valve(pa_1)$                                                             | $se_3 + se_4 \rightarrow pa_1$                                   |
| 2     | $pump(pm), pipe(p_1), pipe(p_6),$<br>$valve(se_3), valve(se_4)$                                                | $se_1 + p_4 \rightarrow se_3$ ,<br>$se_2 + p_5 \rightarrow se_4$ |
| 1     | $pump(pm), pipe(p_1), pipe(p_6),$<br>$valve(se_1), pipe(p_5), valve(se_2),$<br>$pipe(p_4)$                     | $p_2 + v_1 \rightarrow se_1$ ,<br>$p_3 + v_2 \rightarrow se_2$   |
| 0     | $pump(pm), pipe(p_1), pipe(p_6),$<br>$valve(v_1), pipe(p_3), pipe(p_5),$<br>$valve(v_2), pipe(p_2), pipe(p_4)$ |                                                                  |

Table 2: Components in the hierarchical representation for system A.

cedures. The first one (ABSTRACT-OBSERVATIONS) associates the available observations to the abstract levels of the hierarchical representation. Then, the second one (TOP-DOWN-DIAGNOSIS) performs the diagnosis.

In particular, ABSTRACT-OBSERVATIONS takes as input the initially available observations  $OBS_0$ , and, by repeatedly applying the *Abstract* function (corresponding to the *abstract* predicate in Mozetic’s original formulation), determines the available observations (if any) for the more abstract levels. The number associated to the coarsest level with observations available is then stored into variable  $l$ . The levels from  $l$  to 0 are then considered by the procedure TOP-DOWN-DIAGNOSIS: for each level  $i$ , first diagnosis is performed using the *Verify* function (corresponding to Mozetic’s *verify* predicate); then, the diagnoses that are found are translated into their representations at the next more detailed level by the function *Detailed* (corresponding to Mozetic’s *detailed* predicate). Those diagnoses will be the possible candidates for level  $i - 1$ . Moreover, also those candidates (if they exist) which have no abstraction at upper levels, and are thus considered for the first time at the current level, are verified. When level 0 is reached, the final diagnoses are returned. It must be noted that no assumptions are made on the implementation of the *Verify* function (e.g. it can be a generate-and-test method, a GDE-like reasoner, or other MBD technique).

By performing diagnosis at more abstract levels, Mozetic aims at finding those diagnoses that are impossible, thus reducing the number of possible diagnoses, i.e. the size of the search space, at more detailed levels. This makes it possible, in many cases, to perform significantly better than directly solving the most detailed diagnosis problem. Experiments done by Mozetic report, in a medical example using a four-level qualitative model of the heart, a speed up by a factor of 20 over a one-level diagnosis. However, given a hierarchical representation, there are situations where the performance is not as good as one would expect because only part of the hierarchical representation is exploited, as we show in the following.

**Example 1.** Consider the 6-levels hierarchical representation of System A sketched in Table 2, and suppose that the following observations are available: flow at the input of pipe  $p_2$  is  $a > F_k$ , while flow at the output of valve  $v_1$  is  $b < F_k$ , i.e.  $OBS_0 = \{inflow(p_2) = a \wedge outflow(v_1) = b\}$ . Executing ABSTRACT-OBSERVATIONS, the observation at the input of  $p_2$  can be abstracted to level 1 as the input of  $se_1$ , but cannot be abstracted to level 2, where the input of  $p_2$  is not visible anymore. By applying the same line of reasoning, the observation on  $v_1$  is abstracted up to level 2. The TOP-DOWN-DIAGNOSIS procedure can thus start only at level 2.

In this example, diagnostic reasoning considers only the three more detailed levels of the hierarchical representation (no observations can be abstracted higher than level 3), and thus cannot take any advantage from the remaining coarser levels, where reductions of the search space could be in principle obtained with less effort. The result is that the computational savings gained over a plain, one-level diagnosis are not as good as one would expect, because diagnosis is started at an intermediate level of the hierarchy.

In general, when we structurally abstract a diagnosis problem, all the observations that refer only to components that are aggregated become unavailable. Thus, as we climb the hierarchical representation, less observations are available. In the systems where the position of measurement sensors is known and cannot be changed, one can limit this problem by choosing the aggregations in such a way that the most abstract levels have always (or at least often) some observation available. However, in general, whatever hierarchical representation we exploit, as the total number of given observations decreases, the top level from which diagnosis is started tends to become lower, i.e. the number of abstract levels not considered for diagnosis increases. The worst consequence of this problem is that the performance of Mozetic's algorithm can become identical to a plain diagnostic approach. This happens whenever the observations given as input cannot be abstracted to level 1, i.e. when all observations refer only to components that are aggregated in order to obtain level 1. The following example shows one possible scenario in which this situation occurs in System A.

**Example 2.** Suppose that the following observations are available: flow at the output of pipe  $p_3$  is  $a > 0$ , i.e.  $OBS_0 = \{outflow(p_3) = a\}$ . The observation on the output of  $p_3$  cannot be abstracted to level 1, because it is not present at that level. Hence, at the end of the execution of the ABSTRACT-OBSERVATIONS procedure, the most abstract level with observations is level 0. Thus, diagnosis is started at that level.

## 4 Our proposed approach

We propose a technique that aims at improving Mozetic's strategy when structural abstractions are used. Our approach exploits also an additional data structure, called *structural tree* (presented in Section 4.1), that highlights the aggregations performed to build the hierarchical representation.

### 4.1 The structural tree

We associate any hierarchical representation  $H$  (built with structural abstractions) to a *Structural Tree* for  $H$  called  $ST(H)$ . In the tree, each node represents a component of  $H$ , and its sons are its subcomponents.

Given a hierarchical representation of a diagnostic problem  $H = \{(SD_i, OBS_i, COMPS_i)\}_{m,i=0,\dots,m-1}, ST(H)$  is built as follows:

- first, for each component  $f \in COMPS_0$ , a new leaf  $f$  is created;
- then, for each aggregation of  $c_1, \dots, c_k \in COMPS_i$  into  $sc \in COMPS_{i+1}$  such that  $c_1, \dots, c_k \in ST(H)$ , a new node  $sc$  is created such that  $sons(sc) = c_1, \dots, c_k$ .

The root of  $ST(H)$  is associated with the component that represents the whole system, i.e.  $root(ST(H)) = c \in COMPS_{m-1}$ . Figure 2 shows the structural tree for System A which has been obtained from the 6-level hierarchical representation sketched in Table 2.

Since each node  $c$  of  $ST(H)$  is a component  $c \in COMPS_i$  for some  $i$ , we associate to it the theory  $SD_c \subseteq SD_i$  and the observations  $OBS_c \subseteq OBS_i$  (when they are available). Thus, each node  $c$  is associated with a subset of one of the diagnostic problems of  $H$ .

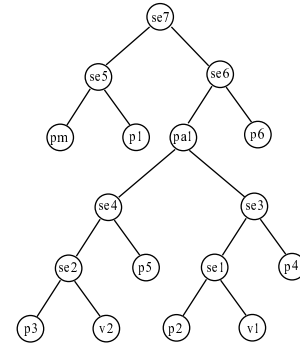


Figure 2: Structural tree for System A derived from the hierarchical representation of Table 2.

### 4.2 The Rearrange algorithm

We propose a technique that dynamically determines a new hierarchical representation suited to diagnose the specific situation described by the current observations. The new hierarchical representation will be built by *rearranging* the levels of a given hierarchical representation  $H$  using the corresponding  $ST(H)$  on the basis of the available observations.

The idea is to improve Mozetic's algorithm when it is not able to fully exploit the hierarchical representation, by providing a new, tailored hierarchical representation consisting of: (i) all levels of the original hierarchical representation which have observations (i.e. the levels considered by Mozetic's algorithm), and (ii) additional, more abstract levels which are not present in the original hierarchy, and have the same observations of the coarsest level considered by Mozetic's algorithm.

We will now show, using an example, how these additional levels can be built with little effort. In some cases, indeed, the abstract level at which diagnosis is started by Mozetic's algorithm is too detailed than necessary. Consider the scenario presented in Example 2, where diagnosis is started at level 0 (which contains 9 components), without exploiting the possibility of starting from a more abstract diagnosis problem. For example, the diagnosis problem (let us call it *Best*) with components  $se5, se3, p3, v2, p5$ , and  $p6$  is more abstract than level 0 but has the same observations (i.e.  $outflow(p_3) = a$ ). The search space of *Best* is much smaller: the diagnosis problem at level 0 has 2304 possible candidates, while *Best* has 288 possible candidates.

Mozetic's algorithm cannot start from *Best* because this diagnosis problem is not available in the given hierarchical representation: the only level with observations is level 0. The problem is that the algorithm exploits the representation in a "fixed" way: levels are identical for any set of observations, and only the choice of starting level changes.

We can derive *Best* by selecting nodes in the structural tree of Figure 2. The structural tree highlights indeed the aggregations used in a hierarchical representation, regardless of the order in which they were performed. One can derive *Best* by selecting a set of nodes in the structural tree which: (i) represent the full system, (ii) have the same observations that are present at level where Mozetic's algorithm starts, (iii) are as abstract as possible. Then, one can compose the theories as-

sociated to the selected nodes to derive a new, more abstract starting level.

We implemented this strategy as an extension to Mozetic's algorithm, which we call the REARRANGE extension. More specifically, the building of the new levels is performed as an additional activity after the ABSTRACT-OBSERVATION procedure, as shown in the following:

Procedure HIERARCHICAL-DIAGNOSIS  
 ABSTRACT-OBSERVATIONS;  
 REARRANGE;  
 TOP-DOWN-DIAGNOSIS.

Procedure REARRANGE

IF  $l < m - 1$  THEN

FOR each node  $n$  of  $ST(H)$  referred in  $OBS_l$

$obs_n \leftarrow \{a = v \mid a = v \in OBS_l \wedge a \text{ is a port of } n\}$ ;

$Nodes_{TL} \leftarrow COMPS_l$ ;

WHILE  $Nodes_{TL} \neq \emptyset$  DO

$n \leftarrow first(Nodes_{TL})$ ;

$f \leftarrow father(n)$ ;

IF  $sons(f) \subseteq COMPS_l \wedge (\bigcup_{s \in sons(f)} obs_s) = obs_f$  THEN

$Nodes_{TL} \leftarrow (Nodes_{TL} - sons(f)) \cup \{f\}$ ;

$l \leftarrow l + 1$ ;

$COMPS_l = (COMPS_{l-1} - sons(f)) \cup \{f\}$ ;

ADD a new level numbered  $l$  with components  $COMPS_l$  to the hierarchical representation  $H$ ;

$Nodes_{TL} \leftarrow COMPS_l$ ;

ELSE

$Nodes_{TL} \leftarrow Nodes_{TL} - sons(f)$ ;

ENDIF;

ENDWHILE;

ENDIF.

The new activities performed by the procedure are:

- each observation available at Mozetic's coarsest level with observations is associated to the corresponding nodes in  $ST(H)$  (obtaining the  $obs_n$  sets);
- if possible, new diagnosis problems, which are more abstract than the one at which Mozetic starts, are added to  $H$ . These new levels are built by:
  1. considering the components of the current, most abstract level (which initially is the original coarsest level with observations);
  2. finding a subset of components that are all the sons of a node  $f$  in  $ST(H)$  such that the union of the observations associated to them are exactly the observations associated to  $f$  (i.e. by substituting them with their father we do not hide any observation);
  3. if such set of components does exist, adding to  $H$  a new level whose components are the components of the current level which are not sons of  $f$ , and  $f$  itself. In this way, we build a more abstract diagnosis problem without losing any observation;
  4. if such set of components does not exist, the algorithm stops; otherwise, the search for components which meet the above described requirements is repeated considering the components of the newly derived level.

After the execution of REARRANGE,  $l$  is the number of the new top level from which diagnosis is started. Note that,

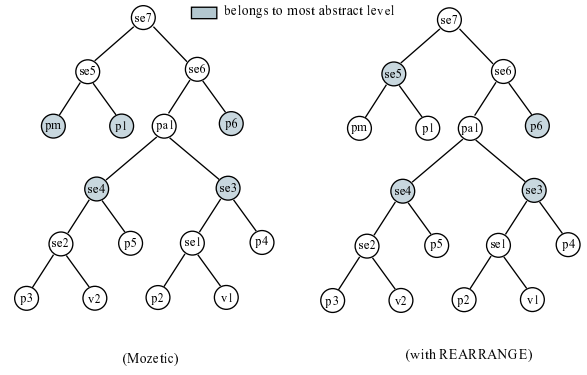


Figure 3: Most abstract levels derived by Mozetic's algorithm and by the REARRANGE extension (Examples 1 and 3).

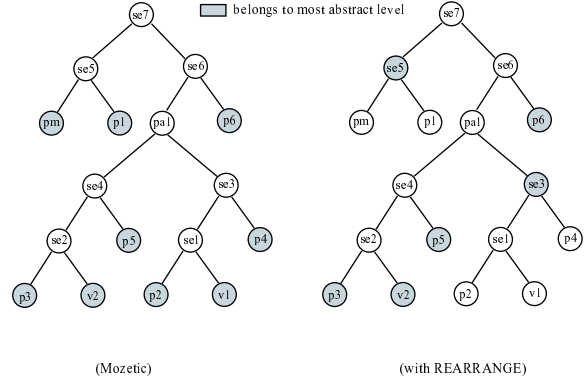


Figure 4: Most abstract levels derived by Mozetic's algorithm and by the REARRANGE extension (Examples 2 and 4).

given the currently available observations, if ABSTRACT-OBSERVATIONS is able to reach the most abstract level, then REARRANGE does nothing, and the whole reasoning proceeds exactly as in Mozetic's algorithm. In order to show the advantages of this strategy with respect to Mozetic's algorithm, we will now reconsider the previously illustrated examples.

**Example 3.** Given the observations in Example 1, Mozetic's hierarchical diagnosis would start at level 2, which contains components  $pm$ ,  $p_1$ ,  $se_3$ ,  $se_4$ , and  $p_6$  (see Figure 3, left). REARRANGE derives an additional, more abstract level 3. It contains components  $se_5$ ,  $se_3$ ,  $se_4$ , and  $p_6$  (see Figure 3, right): components  $pm$  and  $p_1$  of Mozetic's most abstract level with observations are aggregated into pump  $se_5$ . This is possible because  $pm$  and  $p_1$  are the sons of  $se_5$  in the corresponding structural tree (that is, an aggregation for them is available), and the observations associated to them are the same associated to their father (in this case,  $pm$ ,  $p_1$ , and  $se_5$  have no observations). No coarser level can be derived, because there is no node in the structural tree whose sons belong to the newly derived level 3 and that can be aggregated without losing observations. Thus, REARRANGE stops. TOP-DOWN-DIAGNOSIS starts from the newly derived level 3.

**Example 4.** The observations given in Example 2 cannot be abstracted to level 1. Hence, the most abstract level with observations is level 0. By applying our technique, first a new

level is derived with components  $pm, p_1, p_3, se_1, v_2, p_4, p_5$ , and  $p_6$  ( $p_2$  and  $v_1$  can be aggregated without eliminating observations); then, an additional level is derived with components  $pm, p_1, se_3, p_3, v_2, p_5$ , and  $p_6$  ( $se_1$  and  $p_4$  can be aggregated without losing observations); finally, a most abstract level (from which diagnosis will start) is derived with components  $se_5, se_3, p_3, v_2, p_5$  and  $p_6$  ( $pm$  and  $p_1$  can be aggregated without losing observations). The most abstract level used by Mozetic and by our algorithm are graphically illustrated in Figure 4.

## 5 Experimental Evaluation

We experimentally evaluated our extension with respect to the original Mozetic's algorithm. Both algorithms were implemented in SWI Prolog (ISO-compliant free source Prolog) developed by the University of Amsterdam and run on a PowerPC G3 400 Mhz (Apple iMac DV) under the LinuxPPC 2000 operating system. The *Verify* procedure used by both algorithms was implemented as a generate-and-test algorithm, i.e. a procedure that considers each possible candidate and then verifies its consistency with the system description and observations. Obviously, this is the worst choice for implementing the *Verify* procedure, but it gives us a worst case scenario (corresponding to the exact size of the considered portion of the search space).

The experimental comparison was performed with diagnosis problems of different size in the hydraulic domain. The considered systems differed in the layout and in the number of components and ports. More precisely, the simplest considered system was System A (9 components), and the most complex was an Heavy Fuel Oil Transfer System (27 components) of a container ship. For each hydraulic system, we considered several possible sets of observations. Each set contained a number of observations randomly selected between one and  $N_{obs}$ , with  $N_{obs} = trunc(0.4 * n)$  where  $n$  is the number of possible measurements points for the considered system (e.g. in systems with 10 possible measurement points, each set contained between one and four observations). The justification for this choice is twofold: first, with a few observations, the problem is more difficult (and this is the typical case in which one wants to use abstraction); second, in the majority of real cases, observations are very scarce because of cost and reliability of sensors, so considering more observations is not a realistic scenario. Each set with  $k$  observations was generated by assigning a random value to  $k$  measurements points (randomly selected between the available ones). Figure 5 illustrates the average CPU times we obtained with the considered hydraulic systems considering 1000 different sets of observations for each system. As one can see, for both algorithms the average CPU time becomes very high for small increases of the number of components of the considered systems (this is due to the generate-and-test procedure). However, our extension performs significantly better, and is more effective as the size of the considered system increases. This is due to the fact that, in a hierarchical representation with many components and levels, REARRANGE has more opportunities for deriving additional levels, and thus obtaining greater improvements. Finally, the experimental evalua-

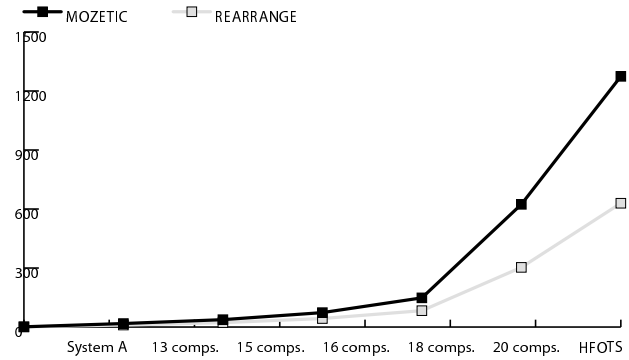


Figure 5: Comparison using 7 hydraulic systems with different number of components.

tion shows that the advantages REARRANGE brings to hierarchical diagnosis outperform the cost of its additional activities.

## 6 Conclusions

We presented an extension to Mozetic's algorithm that improves its performance when structural abstractions are employed. Although improvements were demonstrated experimentally, we definitively need to test our approach with other real-world systems in different domains to evaluate its effectiveness. Moreover, we are also working on alternative rearrangements of hierarchical representations which can be more effective in some situations. Finally, we are considering to include the cost and discrimination power of measurements in the rearrangement process.

## References

- [Autio and Reiter, 1998] K. Autio and R. Reiter. Structural abstraction in model-based diagnosis. In *Proceedings of the Thirteenth European Conference on Artificial Intelligence*, pages 269–273. John Wiley and Sons., 1998.
- [de Kleer et al., 1992] Johan de Kleer, Alan K. Mackworth, and Raymond Reiter. Characterizing diagnoses and systems. *Artificial Intelligence*, 56:197–222, 1992.
- [Genesereth, 1984] M.R. Genesereth. The use of design descriptions in automated diagnosis. *Artificial Intelligence*, 24:411–436, 1984.
- [Hamscher, 1991] W. C. Hamscher. Modeling digital circuits for troubleshooting. *Artificial Intelligence*, 51:223–271, 1991.
- [Mozetic, 1992] Igor Mozetic. Hierarchical diagnosis. In W. Hamscher L. Console, J. de Kleer, editor, *Readings in Model-based Diagnosis*, pages 354–372. Morgan Kaufmann, San Mateo, CA, 1992.
- [Struss, 1992] Peter Struss. What's in sd? towards a theory of diagnosis. In W. Hamscher L. Console, J. de Kleer, editor, *Readings in Model-based Diagnosis*, pages 419–449. Morgan Kaufmann, San Mateo, CA, 1992.