Choosing Abstractions for Hierarchical Diagnosis

Fabien Perrot, Louise Travé-Massuyès

Choosing Abstractions for Hierarchical Diagnosis

Fabien Perrot and **Louise Travé-Massuyès**
{perrot,louise}@laas.fr
0033561336952 0033561336302
LAAS-CNRS
Université de Toulouse
7, avenue du Colonel Roche,
31077 Toulouse Cedex 4
France

Abstract

This paper deals with the choice of abstractions for stating hierarchical diagnosis problems. Generally, hierarchical models are built manually and choosing the appropriate abstractions is quite an empirical science. To tackle this issue, we frame a diagnosis problem as an optimal constraint satisfaction problem (OCSP) and we define abstraction related to two OCSP's, in the structure and in the search space. This allow us to analyse the influence of the abstraction on the temporal computational complexity reduction offered by hierarchical reasoning. Optimal abstractions are shown to be built on the well-known diagnosis concept of potential conflict.

Introduction

In the artificial intelligence community, diagnosis problems are often formulated along the theory of consistency-based diagnosis (Hamscher, Console, & de Kleer 1992), one of the most widely used approaches to model-based diagnosis. However, such a logical formulation is being replaced more and more by a constraint optimisation formulation (Williams & Ragno 2003) which allows one to compare solutions. More precisely, a diagnosis problem is framed as an optimal constraint satisfaction problem (OCSP).

Abstraction has been advocated as one of the main remedies for the computational complexity of model-based diagnosis. It can be viewed as the process of generalisation by reducing the information content of a concept or an observable phenomenon, typically in order to retain only information which is relevant for a particular purpose. Consequently, from a detailed diagnosis problem, one can construct by iterated abstractions more abstract diagnosis problems. These diagnosis problems can be solved together in hierarchical fashion (generally from the most abstract one to the least). Discovered solutions and non-solutions of the i^{th} diagnosis problem can be used to prune the search space of the $i - 1^{th}$ diagnosis problem. Theoretically, temporal computational complexity reduction is exponential. In practice, this reduction notably depends on the choice of the abstract diagnosis problems, and thus depends on the choice of the abstractions. This reduction of complexity is observed in several research fields like symbolic model-checking, planning, constraint solving,.... It can be explained by the fact that abstraction is a way of reasoning about a set of objects (or states) satisfying specific properties rather than directly reasoning on the enumeration of objects. Such sets of states can be proved to be non-solution or solution, thus (if abstraction is well-chosen) it is a proof of solution or non-solution for all the belonging to the sets states. For example, the concept of conflict is by definition a direct proof that a set of states only contains non-solutions. It is more efficient to identify conflicts rather than testing and proving all the states one by one.

The aim of our work is to find, among all possible abstract diagnosis models, the one which gives the greatest complexity reduction. It would allow one to design and implement algorithms that build guaranteed "good" hierarchical diagnosis problems of a system, without the need for expertise. We are especially interested in abstraction techniques for CSP's in order to apply them to diagnosis. Abstraction of CSP's was recently surveyed in (Lecoutre *et al.* 2006) who proposed a new framework to describe general kinds of abstractions. Hierarchical diagnosis based on structural abstraction (which is a special case of abstraction) has been studied in detail in (Chittaro & Ranon 2004). Structural abstraction, from the diagnosis point of view, consists of clustering the components of the system. However, as far as we know, only one publication (Torta & Torasso 2006) in the diagnosis field, has been devoted to the choice of abstractions. The research in (Torta & Torasso 2006) deals with behavioural abstraction [1]. Our approach deals with more general kinds of abstractions.

The rest of this paper is organised as follows. In the second section, we succinctly present the OCSP framework and we recall how consistency-based diagnosis problems can be viewed in that framework. In the third section, we present the hierarchical diagnosis approach and the abstraction concepts. In the fourth and main section, we analyse the influence of the choice of abstractions on temporal computational reduction of hierarchical reasoning. Finally, we conclude and discuss future work.

OCSP's and diagnosis problems

The CSP framework

A constraint satisfaction problem (CSP) is defined as a set of variables and a set of constraints on these variables. Notably,

[1] The authors aggregate component modes but not combinations of modes, a special case of structural abstraction.

it is recognised by the following definition :

Definition 1 *A* CSP *is a pair* (V, C) *where :*

- $V = \{V_1, V_2, \ldots, V_n\}$ *is a set of* variables. *Each variable* V_i *has a non empty* domain $D(V_i)$ *of possible values.*
- $C = \{C_1, C_2, \ldots, C_m\}$ *is a set of* constraints. *Each constraint* C_j *involves some subset* $vars(C_j)$ *of* V *and specifies the allowable combinations of values for that subset.*

A state *of the problem is an assignment to all the variables in* V *:* $(V_1 = v^1, V_2 = v^2, \ldots, V_n = v^n)$. *A* partial state *of the problem is an assignment to only some variables (but not all) in* V. *A* consistent *assignment is one that does not violate any constraints. A* solution *to a CSP is a consistent state. A state is sometimes called a complete assignment and a solution is sometimes called a consistent and complete assignement.*

The structure of a CSP is given by a constraint hypergraph *where the nodes of the hypergraph correspond to variables of the problem and the hyperarcs correspond to constraints.*

The OCSP framework

An OCSP is a CSP for which one is allowed to compare solutions thanks to a cost function; and it is adapted to diagnosis in the sense that one is only interested in values of a subset of variables called decision variables. More formally :

Definition 2 *An* optimal constraint satisfaction problem (OCSP) *consists of a* $CSP = (V, C)$*, a set of* decision variables $W \subset V$*, and a* cost function $g : W \rightarrow \mathbb{R}^2$. *The remaining variables* $V - W$ *are called* non-decision variables. *An assignment of decision variables is called a decision state. A* solution *to an OCSP is a minimum cost decision state that is consistent with the CSP.*

For more details about OCSP's, please refer to (Williams & Ragno 2003).

The diagnosis problem

In the consistency-based diagnosis approach (Hamscher, Console, & de Kleer 1992), a model of the system to be diagnosed is used. This model is component-centered. Each component has a set of possible modes. Behaviour of the system in each mode is described with formulae. Observables are also provided with formulae. Given this model, the task of solving the diagnosis problem consists of finding those modes of the components which are consistent with observations. More formally, the definition of a diagnosis model in the DX community (Hamscher, Console, & de Kleer 1992) is the following :

Definition 3 *A diagnosis model is a triple* $(SD, COMPS, OBS)$ *where :*

- SD *is the system description. It consists of a set of first-order logic formulae which describe the behaviour of the system.*
- $COMPS$ *is a set of constants which represent the components of the system.*
- OBS *is a set of first-order logic formulae which represent the observations given by the sensors.*

The diagnosis problem in the OCSP framework

Like some authors (Williams & Ragno 2003), we argue that a diagnosis problem can be framed as an OCSP (the model) where one must find the n best assignments of mode variables consistent with the constraints describing the system (the task). Decision variables are the mode variables. Solutions are the diagnoses. Non-decision variables are all other variables (including observable and non-observable variables). This has a significant impact because non observable variables may have infinite domains. Considering mode variables as decision variables provides a kind of abstraction that turns an infinite domain CSP into a finite domain CSP. Each mode variable m_{C_i} is attached to one component C_i.

Example The formulation as an OCSP of the diagnosis problem of the classic polybox toy example in the DX community is given in the following.

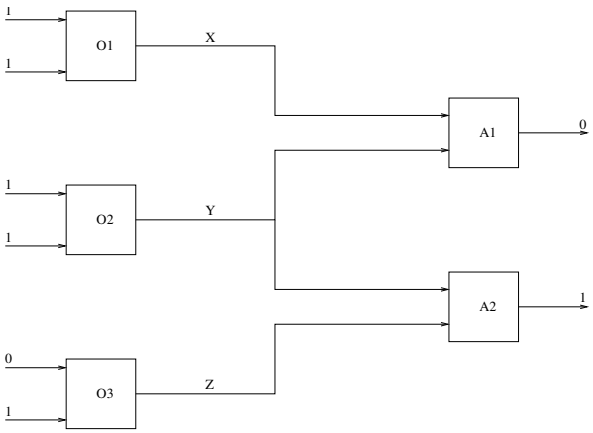

Figure 1: Topology of the boolean polybox

The topology of the polybox is represented in figure 1. O_1, O_2, O_3, A_1, A_2 are the components of the system. O_1, O_2, O_3 are OR gates and A_1, A_2 are AND gates. X, Y, Z are non-observable variables; each of those can take values in $\{0, 1\}$. $m_{O_1}, m_{O_2}, m_{O_3}, m_{A_1}, m_{A_2}$ are mode variables associated to the components; each of those can take values in $\{G, B\}$ (G for Good and B for Bad). A set of constraints link mode variables, observable and non-observable variables. When the observable variable values are provided, one can always instantiate observable variables rather than keeping them in the model. This is why there are no observable variables in the polybox constraint model below.

For sake of clarity we use a constraint language which explicitly describes assignments of values to variables (we could have written the constraints in pure propositional logic). The constraints for the polybox are :

- $(m_{O_1} = G) \implies (X = (1 \vee 1))$
- $(m_{O_2} = G) \implies (Y = (1 \vee 1))$
- $(m_{O_3} = G) \implies (Z = (1 \vee 0))$
- $(m_{A_1} = G) \implies (0 = (X \wedge Y))$
- $(m_{A_2} = G) \implies (1 = (Y \wedge Z))$

These constraints can also be represented in extension because, in this example, the domains of all the involved variables are finite. For instance, the first constraint can be rewritten : $\{(O_1, X), \{(G, 1), (B, 0), (B, 1)\}\}$.

Finally, the diagnosis problem of the polybox in the OCSP framework is :

- decision variables : $\{m_{O_1}, m_{O_2}, m_{O_3}, m_{A_1}, m_{A_2}\}$. Each associated domain is $\{G, B\}$,
- non-decision variables : $\{X, Y, Z\}$. Each associated domain is $\{0, 1\}$,
- constraints are those described above.
- cost function is $g(m_{C_i} = v_i) = \prod_j P_j(m_{C_i})$. Cost represents the candidate probability. The component mode probabilities $P_j(m_{C_i})$ are combined with multiplication because faults on different components are assumed to be independent.

The associated constraint hypergraph is represented in figure 2.

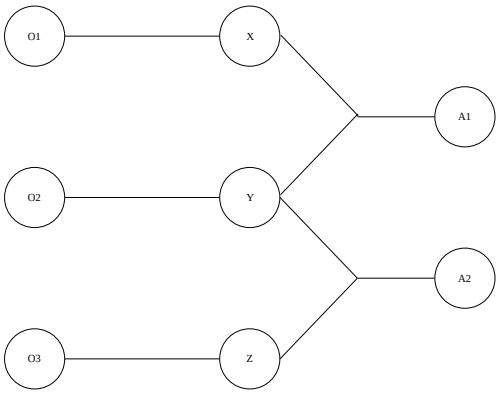

Figure 2: Constraint hypergraph of the boolean polybox

Hierarchical diagnosis and abstraction

Hierarchical diagnosis reasoning

Hierarchical diagnosis reasoning consists of solving a diagnosis problem using an ordered list of diagnosis problems. Generally, this is initiated by a diagnosis problem P_0 and builds up to a more abstract diagnosis problem P_1. Iteratively, one can build an ordered list of n diagnosis problems. Solving these problems is commonly achieved in a top-down fashion. First, the most abstract problem P_{n-1} is solved. Then (or at the same time), knowledge of solutions and non-solutions of P_{n-1} is used (through abstraction) to prune the search space of the problem P_{n-2}. By iterating, one can solve the original diagnosis problem P_0. Results from solving problem P_k can be useful for solving problem P_{k-1} in different ways. It depends on the kind of abstraction which has been used to build problem P_k.

Abstraction concepts

In this section, different views of abstractions are described, each being more or less appropriate to exhibit properties interesting for diagnosis.

Topological view of abstractions (mapping of components) The topological view of abstractions relies on a component-centered representation. Graphically, as shown in figure 3, components are represented by nodes and labeled by their name. The links between components are represented by arcs (direction is chosen according to causality). Each arc is labeled by a value (observable variable) or the name of a non obervable variable. Some abstractions interpret naturally according to the topological view but others cannot be represented in this way. When such a representation exists, topological views of the concrete model and of the abstract model are those giving the best intuition of the abstraction.

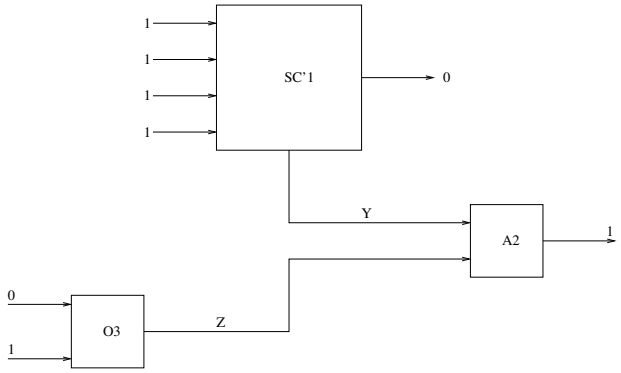

Figure 3: Topology of the abstract boolean polybox.

Example In the polybox example, let us consider a simple structural abstraction (a special case of abstractions) which consists of aggregating for instance the components O_1, O_2 and A_1 in one supercomponent SC'_1. The topology of the abstract polybox is represented in figure 3. But to completely define this abstraction, the mapping between possible values of $(m_{O_1}, m_{O_2}, m_{A_1})$ and $m_{SC'_1}$ must be specified. As an example, a natural mode value mapping is :

- when $(m_{O_1} = G, m_{O_2} = G, m_{A_1} = G)$, then $m_{SC'_1} = G$,
- $m_{SC'_1} = B$ in other cases.

The topological view cannot capture all the abstraction information because the mode value mapping does not explicitely appear.

This mode value mapping is represented in figure 4 and corresponds to the projection on $D(m_{O_1}) \times D(m_{O_2}) \times D(m_{A_1}) \rightarrow D(m_{SC'1})$ of an interpretation mapping as defined in (Nayak & Levy 1995). We can build any value mapping among the 2^8 possible mappings defined on $D(m_{O_1}) \times D(m_{O_2}) \times D(m_{A_1}) \rightarrow D(m_{SC'1})$.

It may be as well the case that mode variables can take more than two values. Some mappings can be more efficient than others, that is discussed in the fourth section. More general kinds of relations than mappings have been deeply described in (Lecoutre *et al.* 2006) but here, just mappings are considered.

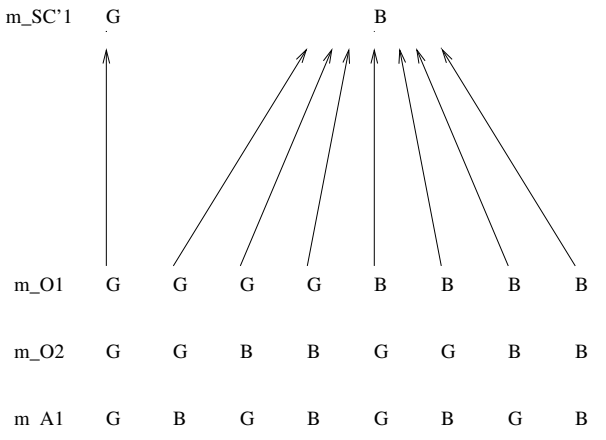

Figure 4: A simple value mapping defined on $D(m_{O_1}) \times D(m_{O_2}) \times D(m_{A_1}) \rightarrow D(m_{SC'1})$.

Limits of the topological view In the last example, one can see that structural abstraction can naturally be defined by two choices :

- aggregation of variables (topological view),
- aggregation of mode value tuples.

However, let us also notice that if we first choose an aggregation of mode value tuples, it implies an aggregation of variables; but the converse does not hold. This remark shows us that criteria for choosing good abstractions, even if it is in the structural abstraction case, can be more easily found by looking directly at the aggregation of mode value tuples (and not at the topological view). This leads us towards a new view of abstractions which corresponds to the search space view.

Search space view (mapping of states) The search space of a diagnosis problem is the set of possible complete assignments to mode variables.

An extended mode value mapping (Lecoutre *et al.* 2006) (Nayak & Levy 1995) associates concrete states to abstract states. This extended mode value mapping can be found extending a mode value mapping to all mode variables. The extended mapping, because its domain spawns on the whole search space, permits to evaluate the number of steps needed by a hierarchical diagnosis algorithm to solve a diagnosis problem. Given that the mode value mapping is generally a surjective mapping (as in the polybox example), the extended mode value mapping is also surjective. Consequently, for each state of the abstract search space, the set of concrete states that correspond to it can be found thanks to the preimage of the extended mapping. The number of elements in this set is called the branching factor of an abstract state. Let us remind to the reader that the preimage of a subset B of the codomain Y under a function f is the subset of the domain X defined by $f^{-1}(B) = \{x \in X | f(x) \in B\}$. The preimage exists even if the mapping is not bijective.

The search space view is difficult to represent by a scheme when the number of components is high because the number of concrete states is exponential in the number of components. In this paper, it is exactly 2^n states for n components. In our polybox example, the 2^5 concrete states are abstracted into 2^3 abstract states by the extended mapping. The latter that we note EM can be found given the mode value mapping M represented in figure 4 in the following manner :

- the domain of EM is $D(m_{O_1}) \times D(m_{O_2}) \times D(m_{O_3}) \times D(m_{A_1}) \times D(m_{A_2}) \rightarrow D(m_{SC'1}) \times D(m_{O_3}) \times D(m_{A_2})$,
- EM maps a concrete state $S = (m_{O_1} = v_1, m_{O_2} = v_2, m_{O_3} = v_3, m_{A_1} = v_4, m_{A_2} = v_5)$ to an abstract state $ES = (m_{SC'1} = M(m_{O_1} = v_1, m_{O_2} = v_2, m_{A_1} = v_4), m_{O_3} = v_3, m_{A_2} = v_5)$.

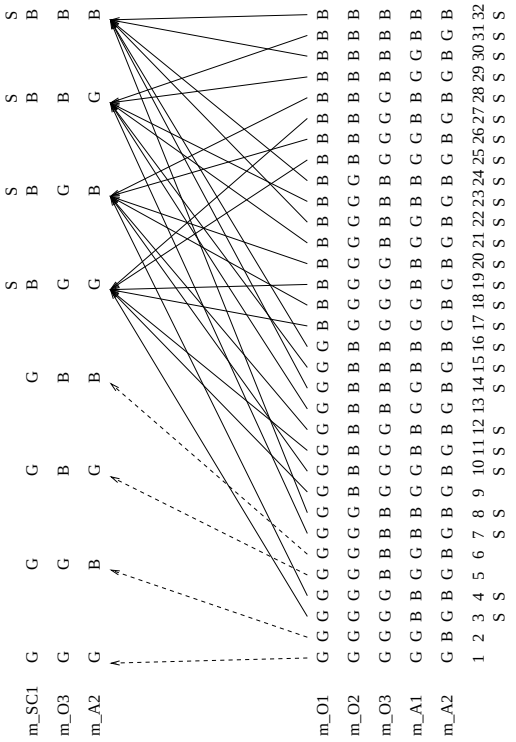

Figure 5: The extended mapping for a structural abstraction of the polybox.

The extended mode value mapping EM is represented in figure 5. Each arrow represents a mapping from a concrete state to an abstract state. The symbol S denotes a state which is solution. One can see that there are 4 solutions to the abstract problem and 26 solutions to the concrete problem.

Constraint view (mapping of constraints) The constraint view of an abstraction is established by the constraint hypergraphs of the concrete and abstract OCSP's.

The constraints of the concrete problem are abstracted into other constraints. In the example of the polybox, constraints 1, 2 and 4 are merged into one constraint : $(m_{SC'_1} = G) \implies ((X = (1 \vee 1)) \wedge (Y = (1 \vee 1)) \wedge (0 = (X \wedge Y)))$. Constraints 3 and 5 are preserved.

The search space view and the constraint view correspond to semantic and syntactic abstractions as defined in (Nayak & Levy 1995), respectively.

Influence of abstractions on computational complexity

The word "best" in "best abstraction" is used for minimal computational temporal complexity, in other words for the minimum number of steps to solve the diagnosis problem. For this analysis, we consider that abstract diagnosis problem(s) are precomputed, so we do not take into account the temporal computational complexity of constructing a hierarchical model of the system to be diagnosed (which is a reasonable hypothesis in the case of on-line state-tracking applications).

The analysis proceeds as follows : firstly, the set of solutions (hence the set of non solutions) is supposed to be known. This permits us to give criteria of "good" abstractions in the space search view; secondly, these criteria are interpreted in the topological and constraint views to determine how they can be used to precompute "good" abstractions that are garantied to be efficient.

Analysis in the search space view

Among all possible abstractions, two general kinds have been identified in the litterature (Nayak & Levy 1995) (Chittaro & Ranon 2004) (Lecoutre *et al.* 2006). :

- Concrete Solution Increasing (CSI),
- Concrete Solution Decreasing (CSD).

Definition 4 (CSI abstraction) *An abstraction is CSI iff for all concrete states which are solutions of the concrete problem, their corresponding abstract state (w.r.t extended mode value mapping) is solution of the abstract problem.*

From this definition, one can trivially deduce the following proposition :

Theorem 1 *Consider a CSI abstraction, then if an abstract state is shown not to be solution, then all its correponding concrete states are not solutions.*

For example, the structural abstraction of the polybox mentioned above is CSI, one can verify it in figure 5.

Definition 5 (CSD abstraction) *An abstraction is CSD iff for all concrete states which are not solutions of the concrete problem, their corresponding abstract state (w.r.t extended mode value mapping) is not solution of the abstract problem.*

From this definition, one can trivially deduce the following proposition :

Theorem 2 *Consider a CSD abstraction, then if an abstract state is shown to be solution, then all its correponding concrete states are solutions.*

A general and simple hierarchical diagnosis algorithm begins from the most abstract problem, generates and tests all the states of a level n in the abstraction hierarchy, then goes down to the level $n-1$, and iteratively, the algorithm finishes to test all the concrete states of the level 0 to give the solutions of the concrete problem.

In this paper, we consider two levels in the abstraction hierarchy but the results can be extended to more than two levels.

Let us recall to the reader that without abstraction, with a simple-minded diagnosis algorithm which just generates and tests states, the temporal computational complexity is $O(n)$ where n is the number of states of the diagnosis problem, i.e. exponential in the number of components.

Consequently, the temporal computational complexity of the hierachical diagnosis algorithm described above depends on two factors :

- the number of abstract states,
- the number of concrete states.

The number of concrete states cannot be changed. An ideal abstraction would consist of two abstract states a and b. a maps all the states which are solutions and b maps all the states which are not. Let us notice that the ideal abstraction we propose is CSI and CSD. So an abstract solution is sufficient to represent all the solutions of the concrete problem; and a non solution abstract state rules out all the non solution concrete states. Consequently, from the CSI and CSD propositions, one can deduce the following proposition :

Theorem 3 *An abstraction that is CSI and CSD reduces the complexity of the diagnosis problem to $O(n')$ where n' is the number of abstract states. The ideal CSI and CSD abstraction has two abstract states.*

But this ideal abstraction cannot be easily built for two reason :

- which states are solutions and which are not is not known in advance,
- constraints associated to the abstract states may not exist.

Refering to the first issue, solutions are supposed to be known (for the analysis) to target which abstractions can easily capture solutions and non solutions. We tackle the second issue using the constraint space view to exhibit abstract constraints which can be easily built.

It appears that the ideal abstraction does not exist in the general case. However one can capture all the solutions and non solutions of a problem using several abstractions of the same problem.

Example

For the polybox example, there are 32 states and, among them, 26 solutions. The diagnosis community is used to represent the 26 solutions by 3 minimal diagnoses : $\{O_1\}, \{A_1\}, \{O_2, A_2\}$. These minimal diagnoses represent 16, 13 and 8 solution states, respectively. Some states are represented by several minimal diagnoses. Non solution states can be captured by minimal conflicts. Minimal conflicts $\{O_1, O_2, A_1\}$ and $\{O_1, A_1, A_2\}$ each captures 4 non solutions.

The ideal abstraction of the polybox example maps the 26 solutions to one abstract state a and the 6 non solutions to another abstract state b. But with such a partition of the search space, building abstract constraints means solving diagnosis problem. However, given that we want to build hierarchical model for all possible set of inputs, this approach is not feasible. One may however notice that using abstract states corresponding to minimal potential conflicts (Cordier

et al. 2004), we can build a relatively efficient hierarchical model for all sets of inputs. All non solutions are captured and ruled out in an efficient way. This idea is developped in the next subection.

Using constraints to find "best" abstractions

We can relax the ideal abstraction requirement using several "two abstract states-based" abstractions to capture the whole search space. One needs, for each abstraction, one abstract state which can represent the highest number of non solutions and another abstract state to represent the highest number of solutions.

To choose these states, the topological and constraint views of abstractions are used. Choosing abstractions in the search space view may lead to non existant abstract CSPs, i.e. for which one cannot find the corresponding variables and constraints.

To represent a maximal number of states, one has two main choices :

- aggregating variables and mode values as described in the topological section,
- removing variables and/or values.

In the constraints view, aggregating variables corresponds to merging constraints and removing variables corresponds to removing constraints. The first option exactly corresponds to structural abstraction and one can see in the polybox example that with natural mappings, abstraction is CSI but does not permit to rule out a lot of non solution states in one test. For the second option, removing n variables permits one to represent 2^n states. So, one has to remove the highest number of variables while keeping the constraints decidable. Removing variables means removing their associated constraints. Consequently, one must find the smallest set of constraints which remain decidable. These sets are hence just overdeterminated, i.e. they involve n equations for $n-1$ unknowns, and they are well-known in the diagnosis field as corresponding to minimal potential conflicts (Pulido & Alonso 2002) (Cordier *et al.* 2004).
When the test on a given OBS of one such just overdeterminated constraint set does not pass, the conjunction of assignments to G of the involved mode variables is a minimal conflict. The case in which at least one variable is assigned to B does not help for finding solutions since the constraints are always satisfied. It is easy to show that when using a minimal potential conflict to construct an abstraction, this abstraction is CSI but not CSD. It is also known that all minimal potential conflicts capture all concrete non solutions. So checking all the abstract states built this way garanties to rule out all the non solution states. However, one abstraction per minimal potential conflict is necessary since one concrete state may correspond to more than one minimal potential conflict. Indeed, we have restricted our analysis to mappings between concrete states and abstract states. With general relations, we conjecture that one abstraction can contain all minimal potential conflicts but in this case, one concrete state may have more than one image.

Conclusion and future work

Hierarchical diagnosis efficiency strongly depends on the choice of the abstractions used to build a hierarchical model of the system. For analysing the influence of this choice, three complementary views of abstractions have been defined : the topological view, the search space view and the constraint view. In the case in which abstractions are built removing variables or contraints, we argue that most efficient abstractions are obtained from minimal potential conflicts because they rule out all non solution states.
But finding efficient abstractions has a cost. For instance, determining minimal potential conflicts is known to be of exponential complexity. So our future work will focus on defining an efficiency measure for an abstraction that takes in account not only the computational gain on the hierarchical model but also the cost of finding the model. In particular, we will pay much attention to structural abstraction because it is generally cheap; and we will consider building a bridge between structural abstractions and other types of abstractions (like conflict-based) that all together may boost the diagnosis resolution process efficiency.

References

Chittaro, L., and Ranon, R. 2004. Hierarchical model-based diagnosis based on structural abstraction. *Artif. Intell.* 155(1-2):147–182.

Cordier, M. O.; Dague, P.; Levy, F.; Montmain, J.; Staroswiecki, M.; and Travé-Massuyès, L. 2004. Conflicts versus analytical redundancy relations : a comparative analysis of the model-based diagnosis approach from the artificial intelligence and automatic control perspectives. *IEEE Transactions on Systems, Man and Cybernetics.* 34(5):2163–2177.

Hamscher, W.; Console, L.; and de Kleer, J. 1992. *Readings in model-based diagnosis*. Morgan Kaufmann.

Lecoutre, C.; Merchez, S.; Boussemart, F.; and Grégoire, E. 2006. A csp abstraction framework. *Revue d'intelligence artificielle* 20(1):31–62.

Nayak, P., and Levy, A. 1995. A semantic theory of abstractions. In Mellish, C., ed., *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 196–203. San Francisco: Morgan Kaufmann.

Pulido, B., and Alonso, C. 2002. Possible conflicts, arrs, and conflicts. *Proceedings of the Thirteenth International Workshop on Principles of Diagnosis (DX 02)*.

Torta, G., and Torasso, P. 2006. Qualitative domain abstractions for time-varying systems: an approach based on reusable abstraction fragments. In *Proc 17th Workshop on Principle of Diagnosis (DX 06)*.

Williams, B. C., and Ragno, R. J. 2003. Conflict-directed a* and its role in model-based embedded systems. *Journal of Discrete Applied Math*.