Diagnosis as a Variable Assignment Problem: A Case Study in Space Robot Fault Diagnosis

Luigi Portinale
Dipartimento di Scienze e Tecnologie Avanzate
Universita' del Piemonte Orientale
Alessandria (ITALY)
e-mail: portinal@al.uniprnn.it

Pietro Torasso
Dipartimento di Informatica
Universita' di Torino
Torino (ITALY)
e-mail: torasso@di.unito.it

Abstract

In the present paper we introduce the notion of Variable Assignment Problem (VAP) as an abstract framework for characterizing diagnosis. Components of the system to be diagnosed are put in correspondence with variables, behavioral modes of the components are the values of the variables and a diagnosis is a variable assignment which explains the observations of the diagnostic problem, by considering the constraints put by the domain theory. In order to have a concise representation of diagnoses and to reduce the search space, we introduce the notion of scenario for representing a set of diagnoses. The paper discusses the definition of preference criteria for ranking solutions and their use for guiding the heuristic search for diagnoses. Experimental data are reported for the evaluation of such a heuristic search on a real-world diagnostic problem, concerning the identification of faults in a space robot arm; in this domain, where a high number of diagnoses may be possible, our approach allows one to get a concise representation of the large number of solutions and to define effective diagnostic strategies able to provide relevant information about fault localization and identification.

1 Introduction

In many real-word applications, diagnostic reasoning is often embedded in a larger task which may involve monitoring, gathering of additional information for hypothesis discrimination, reconfiguration, repair, etc... In such a complex situation, the diagnostic component has to summarize the results of the diagnostic reasoning in such a way that the intelligent agent (either human or artificial) who has to use the diagnostic results is able to perform the task. Unfortunately, for most artifacts the number of possible diagnoses is quite large. In the model-based diagnosis community there is long tradition to use some preference criterion for representing the set of possible diagnoses: the notion of minimal diagnosis [Reiter, 1987] has been often used, but its drawbacks are well known when the domain theory includes also models of the faulty behavior. The characterization of diagnoses in terms of partial and kernel diagnosis [de Kleer *et* al., 1992] is useful, but it does not guarantee at all that the number of diagnoses is small.

In model-based diagnosis a number of approaches based on information theory have been proposed for suggesting additional measurements, in order to discriminate between competing diagnostic hypotheses and to reduce the number of diagnoses. However, it is not always possible to get additional measurements and therefore, in such cases, other techniques have to be developed for representing diagnoses in a more compact way. In several domains fault localization is not sufficient for solving the diagnostic task: fault identification is needed because of different repair actions and/or criticality of the fault. In modeling such domains, the behavior of the diagnosed system is represented in term of behavioral modes and the space of possible solutions is usually quite large. The introduction of a preference criterion among diagnoses is not only useful for representing in a compact way the set of solutions, but it should also guide the search process of diagnoses generation, in such a way that preferred diagnoses are generated before non-preferred ones (see, for example, [de Kleer, 1991] for a probabilistic approach). This is a very important requirement in complex domains where the computation of diagnoses is time (or space) consuming and therefore the computation is time (or space) bounded.

The present paper aims at solving some of the problems mentioned above by introducing a characterization of diagnostic problem solving (in particular abductive diagnosis) as a type of *Variable Assignment Problem* (VAP). In section 2, we introduce the notion of VAP as a problem in which some variables have to be assigned, depending on the constraints induced by some other entities, called findings, through a set of rules. Several kinds of problem solving tasks can be viewed as specific instances of a VAP including planning, diagnosis, classification, learning of operational concepts, etc... In the present paper, we will concentrate on the view of diagnostic problem solving as a VAP: in section 3 we discuss some general issues arising on VAPs, in particular the compact representation of set of solutions (sect. 3.1), the use of information theoretic criteria for preference among solutions (sect. 3.2) and the use of heuristic search for solving VAPs (sect. 3.3). In section 4 we will introduce a real-world diagnostic problem, related to the identification of faults in a space robot arm [Mugnuolo *et* al., 1998] and we will show how mapping such a problem to a VAP can allows us to define diagnostic strategies able to provide the user with a powerful tool for detecting and identifying faults in such a complex system.

2 Variable Assignment Problems

A Variable Assignment Problem or VAP is characterized by a set of *variables* having a finite set of admissible values, and a set of observable parameters (that we will call *findings)* that may constraint the variable values through a suitable set of rules representing the domain theory.

Definition 2.1 *A Variable Assignment Problem (VAP) is a triple (V, F, DT) where:*

- $V = \{x_1, x_2, \ldots x_n\}$ *is a set of* variables *taking values from a predefined set of mutually exclusive values* D_{x_i} *(for* $i = 1 \ldots n$*) called the domain of* x_i*;*

  $F = \{f_1, f_2, \ldots f_m\}$ *is a set of* findings *represented as atomic propositions;*

- $DT = \{R_1, R_2, \ldots R_k\}$ *is a set of rules relating variables and findings and called the* domain theory

In the following, given a variable x_i, we will indicate as $\mathbf{x}_i$ a particular instance of the variable (i.e. an assignment to x_i from $.D_{x_i}$). It should be clear that, since values in $.D_{x_i}$ are mutually exclusive, $\mathbf{x}_i^k \wedge \mathbf{x}_i^h \vdash \perp$ for $(k \neq h)$. Moreover, in the present work we will discuss the case where *DT* is a Horn theory.

A diagnostic problem can be characterized in terms of VAP as follows: each variable corresponds to a *component* of the system to be diagnosed, with each value representing a *behavioral mode* of the component [de Kleer and Williams, 1989]; each finding corresponds to an *observation,* i.e. to an observable parameter of the system; the theory represents the *model* of the system (usually a behavioral model relating behavioral modes of components to observable manifestations). In this paper, for the sake of simplicity, we will ignore the influence of *input* observations in determining the *output* observations and we will restrict our attention to *two-layers* diagnostic problems [Peng and Reggia, 1991], where faults (and possibly normal behavior) are directly related to observations[1].

Solving a VAP means to determine the possible assignments to the variables that "satisfy" the rules concerning the observed findings. This may mean different things: it may be sufficient to require the logical consistency of an assignment with respect to observed findings or it may be the case that a covering of the findings is required. From the diagnosis point of view this corresponds to the classical distinction between *consistency-based diagnosis* and *abductive diagnosis.* In the following we will consider the stronger choice of coverability[2].

Definition 2.2 *A* solution *to a VAP (V.F.DT) is a vector* X = $\langle \mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n \rangle$ *such that* $DT \cup \mathbf{X} \vdash F$.

3 Solving VAPs

3.1 Basic Issues

Given a VAP as defined in def. 2.1, each finding /, constrains the values of a subset of the variables; this means that given f_i,• the set of possible variable assignments is restricted by the rules in *DT* related to f_i

Example 1. Let $\langle \{x_1, x_2\}, \{f_1, f_2\}, DT \rangle$ be a VAP defined on two ternary variables having values $\mathbf{x}_i^j$, i = 1,2; j = 1,2,3 and with the following theory:

$\mathbf{x}_1^1 \rightarrow f_1$; $\mathbf{x}_1^2 \rightarrow f_1$; $\mathbf{x}_2^1 \rightarrow f_2$; $\mathbf{x}_2^2 \rightarrow f_2$

Solving this VAP produces the following variable assignments:

$\langle \mathbf{x}_1^1, \mathbf{x}_2^1 \rangle$; $\langle \mathbf{x}_1^2, \mathbf{x}_2^1 \rangle$; $\langle \mathbf{x}_1^1, \mathbf{x}_2^2 \rangle$; $\langle \mathbf{x}_1^2, \mathbf{x}_2^2 \rangle$

For problem solving efficiency, it may not be reasonable to "expand" all the possible assignments generated by a given finding. By considering one finding at the time, we can notice that each f_i constraints only $\mathbf{x}_i$ via the disjunction $\mathbf{x}_1^1 \vee \mathbf{x}_1^2$, so a more compact representation for the above variable assignments can be the conjunction of disjunctions $(\mathbf{x}_1^1 \vee \mathbf{x}_1^2) \wedge (\mathbf{x}_2^1 \vee \mathbf{x}_2^2)$

In general, we can represent the set of assignments generated by each finding by means of a set of *scenarios,* each being a particular kind of *conjunctive normal form* (CNF) formula.

Definition 3.1 *Given a VAP* $\langle \{x_1, \ldots x_n\}$*,F,DT),* a scenario *is a CNF formula of the type*

$$\bigwedge_{i=1}^{n} (\bigvee_{1 \leq j \leq D_{x_i}} \mathbf{x}_i^j)$$

where each conjunct is a disjunction of (at most D_{x_1}*) instances of the same variable* x_i

In particular, in the case of diagnosis, if the space of elementary variable assignments is very large, a scenario can represent in a compact way multiple assignments with two potential benefits: 1) easier analysis of diagnoses for a human or artificial agent, since they are factored; moreover, remaining sources of indeterminacy requiring further discrimination are pointed out, for each

[1] The whole framework can be generalized to more complex models with arbitrary long chains of rules between behavioral modes of the components (and input parameters) and observations. In such models, variables not corresponding to components but to internal states (endogenous variables) are present. Also these variables have finite domains and the domain theory *DT* contains rules relating input parameters and components with internal variables as well as rules relating internal variables to findings. Input parameters can be viewed as variables whose assignment is already given as an external constraint.

[2] This choice is justified when the model is reasonably complete (see [Console and Torasso, 1991]): as we shall see in section 4, in our case study the behavioral modes as well as their relation to observations are known.

component within each scenario; 2) reduction in the size of the search space. In fact, each finding imposes a constraint on a subset of the variables, represented as a set of *k* admissible tuples for such a subset and this may be summarized in at most *k* scenarios. On the other hand, the adoption of scenarios does not guarantee that search will produce solutions that are elementary assignments, even if they can be easily generated from final scenarios. Therefore, a good reason for adopting scenarios is their ability of trading-off search effort with respect to specificity of assignments. In complex domains, the adoption of scenarios may reduce the search space, but it may still be large (as well as the final set of scenarios): the introduction of a measure of preference among scenarios can then help to focus their generation to most preferred ones.

Example 2. Suppose that the VAP of example 1 is modified by adding to *DT* the following rules:

$\mathbf{x}_1^3 \wedge \mathbf{x}_2^3 \rightarrow f_1$; $\mathbf{x}_1^3 \wedge \mathbf{x}_2^3 \rightarrow f_2$

We have now five variable assignments solving the problem that may be compacted into two scenarios, one very general $(\mathbf{x}_1^1 \vee \mathbf{x}_1^2) \wedge (\mathbf{x}_2^1 \vee \mathbf{x}_2^2)$ and one very specific $\mathbf{x}_1^3 \wedge \mathbf{x}_2^3$;i the problem is how to compare them.

The mechanism we propose to approach this problem is based on the *Minimum Description Length* principle (MDL) [Rissanen, 1983] adapted to the particular problem we are tackling. Next section discusses this topic, while sect. 3.3 addresses the problem of using such a preference information during search.

3.2 Comparing Scenarios

The MDL is a principle of parsimony often used in machine learning or in probabilistic reasoning; the main idea is that hypotheses that may be described more concisely should be preferred over other competing hypotheses. The MDL principle is basically motivated by the fact that, using concepts from information theory, most probable hypotheses have shortest descriptions (see [Mitchell, 1997] for a detailed dis cussion). In particular, given a hypothesis *h* having probability p(h), the optimal encoding for *h* assigns — $\log_2(p(h))$ bits to the hypothesis description. Notice that in our case, the direct encoding of a compound hypothesis represented by a given scenario is not really appropriate; indeed, if a scenario represents the compound hypothesis $h_1 \vee h_2$, the coding length had to be proportional to — $\log_2(p(h_1)) - \log_2(p(h_2))$ which is strictly greater than the coding length of the compound hypothesis given by $-\log_2(p(h_1 \vee h_2))$. In fact, from the information theoretic point of view we want to provide the receiver with the information that there is some indeterminacy in deciding which is the right hypothesis for the variable, so we have to transmit two separate messages: one for describing h_1 and another message for describing h_2.

To apply a comparing principle based on MDL in our case, we first need to introduce some notational facilities. Given a VAP $\langle\{x_1, \ldots x_n\}, \{f_1, \ldots f_m\}, DT\rangle$ we associate with each variable X_i a bitmap B_i of length $l_i = |D_{x_i}|$ such that $B_i[k]$ = 1 if the variable x$_i$, may be assigned the kth value and $B_i[k] = 0$ otherwise. We then define *B* = $B_1 \bullet B_2 \bullet \ldots B_n$ to be a bitmap associated with the whole set of variables, . being the concatenation operator.

It should be clear that every scenario generated by the findings can be encoded as a bitmap *B* defined as above. Of course, a scenario is inconsistent if there exist a B_i in B such that $\forall k B_i[k] = 0$. A bitmap *B* is said to be *elementary* if it represents an elementary variable assignment (i.e. an assignment of exactly one value to each variable x_i), thus if and only if $\forall i = 1..n\ \exists!\ k : B_i[k] = 1$. Every generic bitmap *B* represents one or more elementary bitmap $B^1, B^2, \ldots B^s$ such that $B = B^1|B^2| \quad B^s$ being | the "bitwise or" operator.

In the following, we make the assumption that variables may be assigned independently to each other[3]. Under this assumption, given an elementary bitmap B^j we define its coding length (or equivalently the coding length of the corresponding variable assignment) as:

$$\Gamma(B^j) = -\sum_{k/B_i^j[k]=1} \log_2(p(\mathbf{x}_i^k))$$

The above definition requires the specification of some probability priors on single variable assignments; in case no specific information is available on such assignments we can assume a uniform prior for each variable.

Given a generic bitmap *B* and its elementary bitmaps $B^1, B^2, \ldots B^s$, the coding length of *B* (or equivalently of the corresponding scenario) is then defined as

$$\Gamma(B) = \sum_{j=1}^{s} \Gamma(B^j)$$

In particular, we will be interested in considering different preference criteria among scenarios represented as bitmaps, defined by considering a suitable transform *T(B)* of the bitmap *B* and by computing $\Gamma(\tau(B))$. The choice of r and equation 1 allow us to trade-off the likely hood of a given scenario and its specificity/generality with respect to specific requirements concerning the current task and application. In general, the coding length of each elementary scenario weights the probability of the corresponding assignment, while the sum over the elementary assignments of the scenario gives a penalty to less specific and then less informative ones. The *r* transform allows one to tune the amount of penalty for less informative scenarios, by taking into account the expected use of such scenarios. In particular, for VAPs representing diagnostic problems, we identified the following basic transforms:

T1: $\tau(B) = B$

T2: $\tau(B) = \tau(B_1 \bullet B_2 \bullet \ldots \bullet B_n) = \dot{B}_1 \bullet \dot{B}_2 \bullet \ldots \dot{B}_n$

T3: $\tau(B) = \tau(B_1 \bullet B_2 \bullet \ldots \bullet B_n) = \overline{B}_1 \bullet \overline{B}_2 \bullet \ldots \overline{B}_n$

T4: $\tau(B) = \tau(B_1 \bullet B_2 \bullet \ldots \bullet B_n) = \hat{B}_1 \bullet \hat{B}_2 \bullet \ldots \hat{B}_n$

[3] In diagnostic terms this corresponds to the usual assumption of prior independence among faults.

where $\dot{B}_i$ is the empty bitmap if B_i has all bits set to 1 and $\dot{B}_i$ = B_i otherwise; $\overline{B}_i$ is the bitmap having all zeros, but *in* the most probable position if B_i is a bitmap having all ones and $\overline{B}_i = B_i$ otherwise; B_i is the bitmap obtained from B_i by resetting to zero all but the most probable position.

Scenarios having smaller coding length are then preferred over those having larger coding length. The choice of the proper transform essentially depends on the use of the scenarios representing final solutions. Let us consider a diagnostic setting: in some cases, more specific diagnoses may be more conveniently used because they imply less discrimination effort (i.e. less tests), so Tl may be the most suitable choice; in some other cases the operator using the diagnostic system may be biased towards preferring some particular assignments contained in a final scenario. For example the operator can ignore components for which no restriction on possible behavioral mode has been provided by the observations analyzed so far. This is the basis of the notion of partial and kernel diagnoses [de Kleer *et a*/., 1992]. Transform T2 assures that a component for which all behavioral modes are still possible do not contribute to the evaluation of the F function. Therefore it guarantees that the equivalent (in our framework) of a kernel diagnosis is preferred to a partial (or total) diagnosis generated by the kernel one. Also T3 treats in a different way components for which no information is provided (all the behavioral modes are possible): in such a case, T3 weights the component as it would be assigned the most probable (usually the normal) behavioral mode. It guarantees that the equivalent in our framework of a kernel diagnosis has a coding length *T* not larger of the one of a partial (or total) diagnosis generated by the kernel one. In transform T4 the contribution of each component to the evaluation of T is given by the most probable behavioral mode still admissible: this is equivalent to select the most probable diagnosis among all the elementary diagnoses represented by a final scenario. This preference criterion has usually the effect of preferring the diagnoses with minimum cardinality of faults[4].

A further advantage of the proposed coding length is that it can be computed without determining all the elementary assignments of a given scenario. Given a bitmap *B* let as before B_i be the sub-bitrnaps relative to the single variables $x_{i,}$: let us define a,- to be a coefficient associated with B_i and representing the number of Is in B_i. The following property is trivially verified.

$$\Gamma(B) = -\sum_{i=1}^{n}(\prod_{j\neq i} a_j)(-\sum_{k/B_i[k]=1} \log_2(p(\mathbf{x}_i^k)))$$

It is worth noting that the evaluation of the T function is not expensive since the number of operations involved in the evaluation is $O(n^2)$, where *n* is the number of components (or the number of variables in the VAP problem).

[4] This follows from the fact that normal behavioral mode has usually much higher probability than faulty modes.

In the next section we will show how the coding length of a scenario can be exploited to guide the search for the solutions of a given VAP.

3.3 Heuristic Search

One obvious way for addressing the problem of solving a VAP is to search in the space of all the scenarios generated by the findings. The assumption we made in this work is that findings are processed in a *pipeline* fashion following a specified order $f_1, f_2, \ldots f_m$. This assumption is made for two main reasons: first it simplify the description of the search strategies without loosing generality (the approach can be generalized if this assumption is relaxed), second in many applications this corresponds to a real constraint on the problem (thus findings are only available in such a way or it is necessary to process them in such a way as in many real-time diagnostic applications).

Given a scenario S, the current finding f_j to be considered plays the role of a state-space operator in state-space search: it generates all the scenarios constraining the current one with respect to the rules related to f_j The initial state can then be defined as the trivial scenario $\bigwedge_{i=1}^{n}(\bigvee_{j=1}^{D_{x_i}} \mathbf{x}_i^j)$ corresponding to a bitmap having all bits set to 1.

Example 3. Let us consider a VAP slightly more complex that the one of example 2 in section 3.1: $\langle\{x_1, x_2, x_3\}, \{f_1, f_2, f_3\}, DT\rangle$ involving three ternary variables having values $\mathbf{x}_1^j$, $i = 1,2,3$; $j = 1,2,3$ and with the following theory:

$\mathbf{x}_1^1 \rightarrow f_1$; $\mathbf{x}_1^2 \rightarrow f_1$; $\mathbf{x}_1^3 \wedge \mathbf{x}_2^3 \rightarrow f_1$ $\mathbf{x}_2^1 \rightarrow f_2$; $\mathbf{x}_2^2 \rightarrow f_2$; $\mathbf{x}_1^3 \wedge \mathbf{x}_2^3 \rightarrow f_2$ $\mathbf{x}_3^1 \rightarrow f_3$; $\mathbf{x}_2^2 \wedge \mathbf{x}_3^2 \rightarrow f_3$ $\mathbf{x}_2^2 \wedge \mathbf{x}_3^3 \rightarrow f_3$

The search process starts from the (trivial) initial scenario, $S_0 : (\mathbf{x}_1^1 \vee \mathbf{x}_1^2 \vee \mathbf{x}_1^3) \wedge (\mathbf{x}_2^1 \vee \mathbf{x}_2^2 \vee \mathbf{x}_2^3) \wedge (\mathbf{x}_3^1 \vee \mathbf{x}_3^2 \vee \mathbf{x}_3^3)$ where all the assignments are possible since no finding has been taken into consideration.

By processing the first finding *f* we obtain two successors of S_0: $S_1 : (\mathbf{x}_1^1 \vee \mathbf{x}_1^2) \wedge (\mathbf{x}_2^1 \vee \mathbf{x}_2^2 \vee \mathbf{x}_2^3) \wedge (\mathbf{x}_3^1 \vee \mathbf{x}_3^2 \vee \mathbf{x}_3^3)$ and $S_2 : \mathbf{x}_1^3 \wedge \mathbf{x}_2^3 \wedge (\mathbf{x}_3^1 \vee \mathbf{x}_3^2 \vee \mathbf{x}_3^3)$. In both scenarios variable is unconstrained since there is no relation in theory *DT* between x_3 and f_1. The search process now consider and starting from *S*, only one scenario accounts for both f_1 and f_2, i.e. $S_3 : (\mathbf{x}_1^1 \vee \mathbf{x}_1^2) \wedge (\mathbf{x}_2^1 \vee \mathbf{x}_2^2) \wedge (\mathbf{x}_3^1 \vee \mathbf{x}_3^2 \vee \mathbf{x}_3^3)$. In fact, the explanation of f_2 in terms of $\mathbf{x}_1^3 \wedge \mathbf{x}_2^3$ is inconsistent with the constraints put on *x* in scenario S_1. If we process finding f_2 by taking into consideration scenario S_2, we get just one successor: the scenario S_2 itself. It is worth noting that the explanation of f_2 in terms of $(\mathbf{x}_2^1 \vee \mathbf{x}_2^2)$ is inconsistent with the constraints put by scenario S_2 on x_2. The final step concerns the explanation of f_3; starting from S_3 we got two solutions to VAP represented by scenarios S_4 and S_5 where $S_4 : (\mathbf{x}_1^1 \vee \mathbf{x}_1^2) \wedge (\mathbf{x}_2^1 \vee \mathbf{x}_2^2) \wedge \mathbf{x}_3^1$ and $S_5 : (\mathbf{x}_1^1 \vee \mathbf{x}_1^2) \wedge \mathbf{x}_2^2 \wedge (\mathbf{x}_3^2 \vee \mathbf{x}_3^3)$ Starting from S_2 we get a single solution represented by the scenario $S_6 : \mathbf{x}_1^3 \wedge \mathbf{x}_2^3 \wedge \mathbf{x}_3^1$. In conclusion we have three different solutions (i.e. scenarios S_4, S_5 and S_6) to the VAP; together the three scenarios represents 9 different elementary assignments. Each scenario mentioned before can be represented by the corresponding bitmap; for instance $B(S_0)$ = (111 • 111 • 111) since each of the three ternary variables can assume any value; $B(S_1) = (110 \bullet 111 \bullet 111)$ since there is some constraints on variable x_i, whereas the solution S_4 is represented by the bitmap $B(S_4)$ = (110 • 110 • 100) where all the variables have to satisfy some constraints put by f_1, f_2 and f_3.

We can notice that, because we process one finding at the time, there is a one-to-one correspondence between a finding and a search graph, level, being the jth finding f_j be processed at level *j* (assuming the initial node at level 0).

Since the coding length function *T* defined in section 3.2 allows for a ranking of the solutions, the most natural way of searching through the space of possible scenarios is to define a heuristic search strategy guided by the principle of getting solutions with minimal coding length. In particular, it is possible to devise an *adrnissible* search strategy based on a Best-First Search (BFS). We can define as evaluation function for a generated scenario 5 (with bitmap *B(S))* at search level *j* the function

$$h(B(S)) = \Gamma(\tau(\min(B(S), j)))$$

where $\min(B(S), j)$ is the minimal coding length bitmap that may be obtained from *B(S)* by processing findings from f_{j+1} to f_m. This function is easily computed by considering the set of variables influenced by each finding

fj

Example 4. Consider a problem with 4 binary variables $x_1, \ldots x_4$, 3 findings processed in the order f_1, f_2, f_3 and *r* corresponding to Tl transform. Let the assignments of each variable be equiprobable, but those for X_2, where $p(\mathbf{x}_2^1) < p(\mathbf{x}_2^2)$

Let us assume that the scenario produced after the processing of *f* is

$S_1 : \mathbf{x}_1^1 \wedge (\mathbf{x}_2^1 \vee \mathbf{x}_2^2) \wedge \mathbf{x}_3^2 \wedge (\mathbf{x}_4^1 \vee \mathbf{x}_4^2)$

with findinf_2, f_3 still to be processed and influencing only X_2 and X_3 respectively; the best (minimal coding length) scenario that may be generated from *S* at level 1 is then:

$S_{best} : \mathbf{x}_1^1 \wedge \mathbf{x}_2^2 \wedge \mathbf{x}_3^2 \wedge (\mathbf{x}_4^1 \vee \mathbf{x}_4^2)$

since X_4 will not be changed by f_2, f_3 and x_2 will be set to $\mathbf{x}_2^2$ at the best. Considering a bitmap of length 8 where the ith pair of bits corresponds to $\mathbf{x}_i^1$ and $\mathbf{x}_i^2$ respectively, then $B(S_1)$ = (10* 11*01 *11) and $B(S_{best})$ = (10*0U 01 • 11), so $h(B(S_1)) = \Gamma(B(S_{best}))$

BFS using *h* is *admissible,* since the evaluation function *h* never over-estimates the actual minimal coding length scenario that may be generated from a given search node. If we force the diagnostic system to find not only the single best solution, but several solutions through backtracking, we are guaranteed that the diagnostic system produces solutions (in terms of scenarios) in order of preference (according to the chosen criterion). Since *h* is optimistic in foreseeing the discrimination power of the findings not yet examined, it is common that the coding length of final scenarios (i.e. the solutions) is larger than the one foreseen by expanding intermediate scenarios by using *h.* This means that in some cases *h* is not as informative as we would like and consequently the search space explored by *h* could be large. In order to check the actual applicability of BFS to real-word diagnostic problems, we have performed some tests on a reduced general experimental framework, where general VAPs have been randomly generated and solved by BFS. We have then implemented a random VAP generator, able to produce test sets of problems by setting the following parameters: the number and the cardinality of the variables, the probability distribution on the variable values, the number and the order of examination of findings, the maximum number of scenarios (MAXS) generated by each finding and some random seeds for having different random scenario generations for each finding. Using this VAP generator we have produced five batches of 20 test problems each (for a total of 100 test problems), characterized by 10 variables having cardinality varying from a minimum of 2 to a maximum of 4 values and with a probability distribution over such values ranging from uniforms to very extreme. Each batch was characterized by parameter MAXS varying from 10 to 15 and by a fixed number of findings that in the 5 batches, has been varied from 10 to 20. Average results for each batch are reported on table 1. For each batch we report: the number of findings (NF), the average expansion factor (EF) representing the percentage of the whole search space (in terms of expanded nodes) that has been visited to find the optimum, the average solution factor (SF) representing the percentage of solutions that BFS has been able to find within a time-out of 30 seconds on CPU time, the percentage of time-outs (TO) occurred in the batch and the percentage of cases in the batch for which no solution has been provided within the time- out (NS). The last row reports on the global average. As we can see in more than 20% of the cases BFS is not able to produce a solution within the time-out and in more than 40% of cases it cannot produce ail the solutions. Moreover, we also experimented that as the complexity of the problem increases (essentially in terms of NF and MAXS), BFS is likely to run out of memory without giving any answer. For these reasons, we consider BFS suitable just for relatively simple domains and we focus our attention on alternative strategies based on Greedy Search (GS) with backtracking, where the scenario to be expanded is locally chosen among those generated at the

NF	EF	SF	TO	NOS
10	20.13%	90.61%	9.52%	4.76%
12	41.49%	59.66%	44.44%	22.22%
15	53.36%	90.01%	12.5%	0%
18	39.47%	40.22%	60%	33.33%
20	41.18%	27.5%	80%	40%
AVG	**49.13%**	**61.6%**	**41.29%**	**20.06%**

Table 1: Average results for BFS.

previous step. By resorting to GS we have two general alternatives: still using *h* but in a local way, or directly using the coding length of a scenario as evaluation function. Because of the lack of space, we cannot describe in detail the experiments with GS using the random VAP generator. However, it is worth noting that the analysis with the VAP generator showed more benefits in directly using the coding length *T* as evaluation function. In the next section, we report on experimental results on the performance of GS on a real-world diagnostic problem.

4 The SPIDER Case Study

In this section we report on a recent work done inside the project SISRAS (Italian acronym for "An Intelligent System for Supervising Autonomous Spatial Robots") sponsored by ASI (the Italian Space Agency) aimed at demonstrating the feasibility of interactive autonomy for controlling and supervising a complex system in the space. The test-bed of the project is the space robot arm SPIDER (Space Inspection Device for Extravehicular Repairs) IMugnuolo *et a*/., 1998]. SPIDER is a 7 degrees of freedom (i.e. a 7 joints) space robot arm developed inside the JERICO (Joint European Robot In-orbit Calibration and Operations) project. In the multi-agent architecture devoted to supervising SPIDER, one diagnostic agent is responsible to identify failures and malfunctions during SPIDER activity, by analyzing symptoms obtained via monitoring of the arm and to provide the human operator with a concise and comprehensible description of the possible faults.

We have devised a behavioral model for SPIDER characterized by 33 components with an average number of 5 behavioral modes each (ranging from a minimum of 2 to a maximum of 9) and 45 observable parameters (among which 16 not equipped with a sensor for direct observation) by taking into consideration the FMECA documents developed during the design and test of the SPIDER arm. Such documents list all the faults for each component and provides a short characterization of each fault in terms of observables. In this way the model we have developed is complete in terms of faults and is reasonably complete in terms of relations between behavioral modes and observations. Since several faults share the same set of symptoms with no possibility of discrimination through monitoring parameters, a very large number of diagnoses may be produced, even when all observable parameters are available and actually observed. We have then approached this problem by considering the diagnostic problem as a VAP; this allowed us to take into account all the considerations we made in previous sections and in particular:

- the processing of observations in the order provided by the monitoring unit;
- an abductive characterization of the diagnostic process, because of the possibility of obtaining a (almost) complete model of both the normal and abnormal behavior of the arm;
- the compact representation of a set of elementary diagnoses through the notion of scenario;
- the definition of diagnostic strategies viewed as heuristic search in the space of possible scenarios;
- the definition of preference criteria for diagnoses obtained as final scenarios at the end of the process.

	IF	EF	O1	O4	DO	TO
T1	1	28.51%	84	96	0.285	0
T1	2	17.53%	61	82	0.289	2
T2	1	27.03%	97	99	0.009	0
T2	2	15.77%	91	97	0.005	0
T3	1	27.03%	97	99	0.065	0
T3	2	16.05%	88	96	0.056	2

Table 2: Results for GS in the SPIDER domain.

Actually the diagnostic system we have developed is more complex than the characterization of diagnosis we have described in previous paragraphs: in particular, the system has to deal with input parameters representing the predicted status of the arm joints (e.g. whether the current command executed by the robot control involves a movement of the joint). As mentioned before, input parameters are dealt with as external constraints on the possible behavioral modes of components; a more detailed description of the modeling issues involved in diagnosing SPIDER (including the exploitation of the dependencies among findings) is reported in [Portinale *et a*/., 1999].

As pointed out in section 3.3, we adopted a GS approach guided by the coding length function defined in section 3.2, also used to address the problem of preference among diagnoses. We have implemented a simulator on the behavioral model of SPIDER able to generate diagnostic cases by injecting faults in the model. In the present paper we report on some of the experiments performed so far: if particular we considered two test sets of 100 cases each, by injecting one and two faults respectively. Each test set has then been tested by running a GS algorithm guided by T under T I , T2 and T3 transform. The average number of observations to be explained in each case was about 19 and a time-out of a 1 minute CPU time has been set. Table 2 summarizes the results in terms of number of injected faults (IF), average expansion factor (EF), number of times where optimum is the first solution (01), number of times where optimum is in the first 4 solutions (04), average distance of the coding length of first solution with respect to the optimum (DO) normalized in [0,1] with respect to the maximum value, number of time-outs (TO). First two lines of table 2 refer to transform T I , next two lines to transform T2, while the last two to transform T3. As we can see, more complex problems (i.e. those involving two faults) are more likely to be timed-out under the time constraint we set up[5]; however, the performance of

[5] Notice that the time-out refers to the algorithm searching

the algorithm appears to be very good, both in quantitative (e.g. EF) and in qualitative terms (e.g. 0 1 , 04 and DO). In particular, it is worth noting that very often GS is able to get the optimum as a first solution (or at least in the first 4); moreover even when the optimum is not obtained as a first solution, the quality of such a first solution is very high as suggested by reported values on DO. This is particularly true for both transform T2 and T3 (with T3 being slightly better). The use of a greedy strategy guided by *T* seems then to be a very effective approach for domains, like SPIDER, where the complexity of the model and the large number of possible solutions have to be properly addressed.

5 Discussion

In the present paper, we have discussed an approach to diagnosis based on viewing a diagnostic problem as a variable assignment problem, where variables (i.e. system components) are indirectly constrained through other observable entities (i.e. system observations). In particular, we addressed the problem of searching for solutions (i.e. diagnoses) in a large solution space, by proposing heuristic search in the space of scenarios (i.e. CNF formulae representing multiple elementary diagnoses). Such a characterization seems to be quite promising in domains where observable parameters do not allow in general a precise discrimination between diagnostic hypotheses (so, we have a large number of competing diagnoses) and it is not possible to get additional measurements. These characteristics are present in the SPIDER domain, but are not exclusive of such a domain: several other real-world applications can have the same problems. Experimental results show that even the use of non-admissible search algorithms based on a. greedy strategy can provide interesting results, especially concerning the production of the best scenario with respect to the given preference criterion.

Characterization of diagnosis as variable assignment has some similarities with work on diagnosis as constraint propagation as discussed in [ElFattah and Dechter, 1995]. The main differences concern the fact that we are focusing on abductive diagnosis on causal/behavioral models rather than on consistency-based diagnosis on structural/behavioral models; moreover, modeling the type of diagnostic problems we discussed here with the dual graph technique proposed in [ElFattah and Dechter, 1995] is likely to produce complex cycles in the graph, making the problem computationally hard in general.

Strategies based on heuristic search (in particular BFS) have also been proposed both in logical (consistency-based) [de Kleer, 1991] and in probabilistic characterizations [Biswas *et al,* 1997; Peng and Reggia, 1991]; in both cases (even when more sophisticated bayesian methods are applied), search is performed on the space of elementary assignments of behavioral modes to components, potentially producing an explosion of the number of possibilities to be examined; our approach aims at avoiding this by means of scenarios.

Acknowledgments

The research described in this paper has been funded by ASI under grant n. 155/97. Partners in the project are IP-CNR, IRST-ITC, Universities of Genova, Palermo, Parma, Roma and Torino. We thank GianlucaCorrendo for his valuable work in developing the prototype of the diagnostic system.

References

[Biswas *et a*/., 1997] G. Biswas, R. Kapadia, and X.W. Yu. Combined qualitative-quantitative steady-state diagnosis of continuous-valued systems. *IEEE Trxms. SMC,* 27(2): 167-185, 1997.

[Console and Torasso, 1991] L. Console and P. Torasso. A spectrum of logical definitions of model-based diagnosis. *Computational Intelligence,* 7(3): 133-141, 1991.

[de Kleer and Williams, 1989] J. de Kleer and B.C. Williams. Diagnosis with behavioral modes. In *Proc. 11th 1JCA1,* pages 1324 1330, Detroit, 1989.

[de Kleer *et a*/., 1992] .). de Kleer, A. Mackworth, and R. Reiter. Characterizing diagnoses and systems. *Artificial Intelligence,* 56(2˙3):197-222, 1992.

[de Kleer, 199l] J. de Kleer. Focusing on probable diagnoses. In *Proc. AAAI 91,* pages 842-848, Anaheim, CA, 1991.

[ElFattah and Dechter, 1995] Y. ElFattah and R. Dechter. Diagnosing tree-decomposable circuits. In *Proc. IJCAI 95,* pages 1742 1748, Montreal, 1995.

[Mitchell, 1997] T. Mitchell. *Machine L carntnq.* Mc Graw Hill, 1997.

[Mugnuolo *et ai,* 1998] R. Mugnuolo, S. Di Pippo, P.G. Magnani, and E. Re. The SPIDER manipulation system (SMS). *Robotics and Autonomous Systems,* 23(1-2):79 88, 1998.

[Peng and Reggia, 1991] Y. Peng and .). Reggia. *Abductive inference models for diagnostic problem solving.* Springer-Verlag, 1991.

[Portinale *et al,* 1999] L. Portinale, P. Torasso, and G.L. Correndo. Knowledge representation and reasoning for fault identification in a space robot arm. In *Proc. 5th Int. Symp. on A I, Robotics and Automation in Space,* Noordwijk, 1999.

[Reiter, 1987] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence,* 32(1):57 96, 1987.

[Rissanen, 1983] J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics,* 11(2):416-431, 1983.

for all the solutions; in the examples we tested the algorithm is always able to find at least one solution within the timeout.