Identifying Model-Based Reconfiguration Goals through Functional Deficiencies

Emmanuel Benazera[1] and **Louise Travé-Massuyès**[2]

Abstract. Modern monitoring and supervising systems track several of the potential system discrete and continuous (hybrid) states in parallel. This results in a difficulty to act when a system is diagnosed to be faulty while its state is not uniquely determined. Another consequence is that the functionalities that have been lost by the system are undetermined and cannot be efficiently restored. We propose to extract the functional deficiencies from a nominal prediction of the system expected states and the belief state that results from the diagnosis operation. We give a characterization of the deficiencies whose size is minimal, while deficient over the largest number of state estimates. Interestingly, functional deficiencies do not overlap. We then identify the reconfiguration goals by making use of a conditioned causal representation of the system equations.

1 Introduction

Model-based autonomous systems already face faulty situations with some success: they detect and diagnose faults by either identifying potential candidates for their own physical state [8] or reasoning on their structural and behavioral knowledge [6]. The next step toward more autonomy is to have the system recovering itself after faults occur, a process known as *model-based reconfiguration*[3] (MBReconf). Automated reconfiguration comprehends three steps: goal identification, goal selection, recovery. *Goal identification* searches for a set of potential states of the system in which the fault is inhibited; *goal selection* is the process of deciding the best of these states, denoted goal states; *recovery* searches for the chain of actions that may turn the physical system state into the desired goal states. Due to several factors, MBReconf is a challenging problem:

- The state of the system cannot be uniquely determined in all situations. Recent model-based monitoring/diagnosis systems track several potential non-faulty/faulty state estimates simultaneously [12, 8, 9, 2].
- Fault effects may differ from one state estimate to the other. For this reason, pre-compiled policies may fail recovering the system by triggering an improper command when the state is uncertain.
- Nowadays, embedded digitally controlled systems have complex behaviors characterized by a preeminence of discrete switches in their dynamics. Due to potential automated switches, there exists no trivial mapping from faults to reconfiguration goal states.

A fault alters some variables of the system state so some functionalities do not appear to be achieved anymore. Thus, a stuck closed valve in a propulsion system loses the thrust functionality. A resistor parameter change alters a thermostat's heating functionality. Here a functionality of the system is described as a conjunction of variable instances. The uncertainty over the system state precludes an easy identification of the *functional deficiencies*. Referring to the *faulty states* as the estimates that result from the diagnosis operation, as opposed to the nominally *predicted states*, we propose to proceed to a comparison to determine the functional deficiencies caused by the faults. In this context, functional deficiencies are variable instances in one or more predicted states that have been *lost* in one or more faulty states. Our approach aims at minimizing the size of a deficiency to recover while maximizing its coverage of the state estimates.

Then, due to the absence of a bijective mapping between the system modes and continuous regions of behavior, it is difficult to identify potential reconfiguration goals based on the functional deficiencies. A solution is to call for reachability analysis [3] to find the system states where functional deficiencies are restored. Unfortunately this time analysis is much too computationally expensive to be integrated into the model-based diagnosis and reconfiguration loop. Instead we propose a methodology based on a conditional causal representation of the state equations, thus abstracting the time away. Reconfiguration then becomes the problem of determining conditions that are sufficient to overcome a functional deficiency. This process is similar to the model-based diagnosis consistency approach [5]. Section 2 details the hybrid framework; section 3 defines and characterizes the functional deficiencies; section 4 identifies the reconfiguration goals.

2 Hybrid Model-Based State Prediction and Diagnosis

In this section we introduce a comprehensive formalization of model, state and uncertainty. The autonomous system is considered a model-based system, i.e. that has a structural and behavioral knowledge of itself.

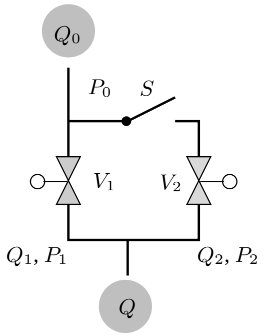
Definition 1 (Model-Based System). *A model-based system A is a tuple $(\mathcal{C}, \mathcal{M}, \mathcal{T}, \mathcal{X}, E)$, where $\mathcal{C}$ is a set of modeled components, $\mathcal{M}$ a set of finite discrete variables as component behavioral modes, $\mathcal{T}$ a set of transitions among these modes, $\mathcal{X}$ the set of continuous variables partitionned in state variables $\mathcal{X}_X$, output (observed) variables $\mathcal{X}_Y$ and input variables (commands) $\mathcal{X}_U$. E is a set of continuous static/differential equations over $\mathcal{X}$.*

The physical system state description is hybrid: the *hybrid state* s is the tuple (M, X). Instances of variables v in $M \cup X$ are noted

[1] RIACS/NASA Ames Research Center, Moffett Field, California 94035 email: ebenazer@email.arc.nasa.gov

[2] LAAS-CNRS, 7, av. du Colonel Roche, 31077 Toulouse Cedex 4 email: louise@laas.fr

[3] For now, most embedded controllers include pre-compiled recovery policies as part of a rule-based system.



$$V_i \left\{ \begin{array}{l} Q_i = k_i S_i \sqrt{P_0 - P_i} \textbf{ if } \phi_i \\ Q_i = 0 \textbf{ if } \neg\phi_i \\ \phi_i\text{: } P_0 \geq P_i \\ \quad \wedge (V_i.m = open \vee V_i.m = stuck\_open) \\ \\ \tau_{11}\text{: } V_1.m = closed \wedge V_1.cmd = op \rightarrow V_1.m = open \\ \tau_{12}\text{: } V_1.m = open \wedge V_1.cmd = cl \rightarrow V_1.m = closed \\ \tau_{21}\text{: } V_2.m = closed \wedge \textbf{S.m=closed} \rightarrow V_2.m = open \\ \tau_{22}\text{: } V_2.m = open \wedge \textbf{S.m=open} \rightarrow V_2.m = closed \end{array} \right.$$

$$S \left\{ \begin{array}{l} \tau_3\text{: } S.m = open \wedge \mathbf{(}P_0 \geq P^*\mathbf{)} \rightarrow S.m = closed \\ \tau_4\text{: } S.m = closed \wedge \mathbf{(}P_0 < P^*\mathbf{)} \rightarrow S.m = open \end{array} \right.$$

Input connection: $Q_0 = Q_1 + Q_2$

$$\text{Output connection: } \left\{ \begin{array}{l} Q = Q_1 + Q_2 \\ P_1 = P_{atm} \\ P_2 = P_{atm} \end{array} \right.$$

Figure 1. Pressure expansion system

$(v = v^j)$, or v^j for short. The hybrid state's discrete side abstracts the physical system as a set of mode instances $M = \bigwedge_k C_k.m^{i_k}$ where $C_k.m^{i_k}$ is an instance of a variable $m \in \mathcal{M}$ of component $C_k \in \mathcal{C}$. The continuous state X is made of instances x^j of continuous variables of $\mathcal{X}_X$. Observed variables of $\mathcal{X}_Y$ are noted y (vector Y), and $\tilde{y}$ (vector $\tilde{Y}$) denotes the measured value. Commands are noted u (vector U). System A's behavior is described with rules of the form $(\bigwedge_i e_i$ **if** $\phi)$, where $e_i \in E$ and ϕ is a conjunction of equalities/inequalities over functions of variables in $\mathcal{M} \cup \mathcal{X}$. A set $T = \{\tau_1, \cdots, \tau_{n_m}\}$ of transitions is specified for each mode m. Each transition τ is enabled according to a guard ϕ, and may trigger with probability $p(\tau)$ whenever the guard is satisfied. $T(s_i, s_j)$ denotes the set of transitions that moves A from s_i to s_j.

We note $\mathcal{P}(A)$ the prediction of the system's nominal state, and $\mathcal{D}(A)$ the diagnosis result after a fault occurs. We denote $\mathcal{S} = \big(\mathcal{P}(A), \mathcal{D}(A)\big)$.

Example (Pressure expansion system). *Figure 1 pictures our case study: a two valves system that limits water pressure between flow input Q_0 and flow output Q. An electric switch S powers valve V_2 when pressure P_0 equals or exceeds threshold P^*. V_2 opens when powered. S, V_1 and V_2 have two nominal operational modes* open *and* closed, *and two faulty modes* stuck_closed, stuck_open. *Q_0 and Q are measured. $P_0 \geq P_{atm}$ is the uncertain input to the system. P_{atm} denotes the atmospheric pressure.*

Our scenario assumes faults occur when the prediction of the nominal state is uncertain[4], i.e. the uncertainty on the pressure does not allow to discriminate between two predicted states in $\mathcal{P}(A)$[5]:

$$s_N^1 : \left\{ \begin{array}{c} Q_0 > 0, P_0 < P^* \\ V_1.m = open \\ S.m = open \\ V_2.m = closed \\ Q_1 > 0, Q_2 = 0, Q > 0 \end{array} \right. \quad \text{and } s_N^2 : \left\{ \begin{array}{c} Q_0 > 0, P_0 \geq P^* \\ V_1.m = open \\ S.m = closed \\ V_2.m = open \\ Q_1 > 0, Q_2 > 0, Q > 0 \end{array} \right.$$

After observing $Q_0 > 0 \wedge Q = 0$, based on the knowledge of the nominal states above, $\mathcal{D}(A)$ is:

$$s_F^1 : \left\{ \begin{array}{c} Q_0 > 0, P_0 < P^* \\ \mathbf{V_1.m = stuck\_closed} \\ S.m = open \\ V_2.m = closed \\ Q_1 = 0, Q_2 = 0, Q = 0 \end{array} \right. , s_F^2 : \left\{ \begin{array}{c} Q_0 > 0, P_0 \geq P^* \\ \mathbf{V_1.m = stuck\_closed} \\ S.m = closed \\ \mathbf{V_2.m = stuck\_closed} \\ Q_1 = 0, Q_2 = 0, Q = 0 \end{array} \right.$$

$$\text{and } s_F^3 : \left\{ \begin{array}{c} Q_0 > 0, P_0 \geq P^* \\ \mathbf{V_1.m = stuck\_closed} \\ \mathbf{S.m = stuck\_open} \\ V_2.m = closed \\ Q_1 = 0, Q_2 = 0, Q = 0 \end{array} \right.$$

s_F^1 is the faulty state diagnosed from s_N^1 while s_F^2 and s_F^3 have been deduced from s_N^2. Hybrid states in $\mathcal{P}(A) = (s_N^1, s_N^2)$ and $\mathcal{D}(A) = (s_F^1, s_F^2, s_F^3)$ contain enough information for extracting the system's *functional deficiencies*.

3 Functional Deficiencies

Variable deficiencies affect the functionalities of the system, i.e. roles it has to accomplish, some being lost or degraded. Functionalities have no precise characterization besides those given by the engineers. In our case, we consider such characterizations are not specified in the system model. Our first objective is to make lost functionalities apparent given the set of potential faults.

Given a belief on a model-based system A, we extend $\mathcal{P}(A)$ and $\mathcal{D}(A)$ by the states probabilities such that $\mathcal{P}(A) = ((s_N^1, p(s_N^1)), \cdots, (s_N^n, p(s_N^n)))$ is the set of the n nominally predicted states, and their associated probabilities, and $\mathcal{D}(A) = ((s_F^1, p(s_F^1)), \cdots, (s_F^f, p(s_F^f)))$ the set of f faulty states from diagnosis, and their attached probabilities. Given a variable v, we note $s(v)$ its value in state s. Any set of nominal and faulty states in $\mathcal{S}$ is denoted a *reconfiguration set*.

We want to find sets F_i of variable instances in $M \cup X$ that characterize the differences between states in $\mathcal{P}(A)$ and $\mathcal{D}(A)$. We show that the minimal functionalities that cover the maximum number of state estimates are a group of non-intersecting F_i. The general idea that is developed in this section has been inspired by the model-based reconfiguration of logical functions in [14].

3.1 Deficient variable instances

Given two states (s_N, s_F) respectively from $\mathcal{P}(A)$ and $\mathcal{D}(A)$, and a variable v, we note $L\big(s_N(v), s_F(v)\big)$ the measure of the common ground of v's value in each state. We say that variables whose instances in a pair of nominal/faulty states have less common ground than observable variables that discriminate between these two states, are *deficient*. We write that v is deficient if:

$$L\big(s_N(v), s_F(v)\big) \leq \max_{y \in Y_{misb}} L\big(s_N(y), s_F(y)\big) \qquad (1)$$

where $nbr(Y_{misb})$ is the number of *misbehaving* observed variables. A misbehaving y is an observed variable that is reponsible for the fault detection, thus discriminating s_N from s_F: y's value in s_F better fits $\tilde{y}$ than its value in s_N. When relation (1) is satisfied, we say

[4] This corresponds to the general case of tracking multiple states simultaneously.

[5] Flows > 0 are abstracted from their real values for an improved readability.

$L\big(s_N(v), s_F(v)\big)$ is deficient. The expression of L and the misbehaving variables depend on the nature of the variables and the formalization of the uncertainty in the model.

In the case variable domains are discrete, as in [16], variable instances have attached boolean labels. Misbehaving variables are observables labeled 1 in s_N and 0 in s_F. We set up $L\big(s_N(v), s_F(v)\big) = 1 - \big(lab(s_N(v)) - lab(s_F(v)\big)$, where lab returns the label of a given instance. This case also applies to the measure of mode deficiencies.

In case variable instances are numerical intervals, as in [2], a misbehaving observed variable y is such that $s_N(y) \cap \tilde{y} = \emptyset$. We use $L\big(s_N(v), s_F(v)\big) = s_N(v) \cap s_F(v)$.

In case a variable estimate is represented with a Gaussian, as in [9], we say y is misbehaving if $p(\tilde{y} \mid s_F)p\big(T(s_N, s_F)\big) \geq p(\tilde{y} \mid s_N)$, i.e. if its likelihood is higher in the diagnosed estimate than in the nominally predicted one, given the probability of changing mode. Here $p\big(T(s_N, s_F)\big) = p\big(s_N(\phi_1, \cdots, \phi_r)\big) \prod_{i=1,\cdots,r} p(\tau_i)$ where r is the number of components, transiting from s_N to s_F. Given that $s_N \sim \mathcal{N}(m_N, \theta_N)$ and $s_F \sim \mathcal{N}(m_F, \theta_F)$, we define L as the measure of the common space enclosed by both density functions f_N, f_F. There exist several ways to assess for this value, one common measure is the Kullback-Leibler divergence.

3.2 Functional Deficiencies

Based on deficient variables, we can build the functional deficiencies.

Definition 2 (Functional deficiency). *A functional deficiency F for a model-based system A over a set of hybrid states $\mathcal{S} = \big(\mathcal{P}(A), \mathcal{D}(A)\big)$ is a set of variable instances of $M \cup X$ that hold in some states of $\mathcal{P}(A)$, and that are* deficient *in some states of $\mathcal{D}(A)$. We denote as $S(F) \in \mathcal{S}$ the reconfiguration set associated to F, $S_N(F)$ and $S_F(F)$ the corresponding sets of nominal and faulty states in $S(F)$, respectively; i.e. $S(F) = (S_N(F), S_F(F))$.*

We write F as a conjunction of n_m mode instances and n_c probabilized value instances, $n_m + n_c = n$, as follows:

$$F = \bigwedge_{k=1,\cdots,n_m} C_k.m^{h_k} \bigwedge_{j=1,\cdots,n_c} \big(\sum_{i=1,\cdots,p} p(s_N^i) s_N^i(v^j)\big) \quad (2)$$

Then $(s_N^i, s_F^l) \in S(F)$ iff: $L\big(s_N^i(C_k.m^{h_k}), s_F^l(C_k.m^{h_k})\big)$ and $L\big(s_N^i(v^j), s_F^l(v^j)\big)$ are deficient for all i, j, k, l. In other words, $S(F)$ includes all nominal and faulty states whose pairs show a deficiency for all the instances of F. F is said to be *complete* w.r.t. a reconfiguration set S' iff $S' = S(F)$. The complete F over $\mathcal{S}$ is unique.

Property 1. *If F, F' are complete functional deficiencies, then if $F' \subseteq F$, $S(F) \subseteq S(F')$.*

Proof. If $F' \subseteq F$, then $S(F')$ contains at least all states of $S(F)$ as these show deficiencies for all instances of F, plus potential states that do not show deficiencies for instances in $F \setminus F'$. □

Property 2. *If F, F' are complete functional deficiencies and $S(F) = S(F')$, then $F = F'$.*

Proof. This comes from the uniqueness of a complete functional deficiency over a given reconfiguration set S. □

Given two tuples $\big(F_1, S(F_1)\big)$ and $\big(F_2, S(F_2)\big)$, we write:

$$\big(F_1, S(F_1)\big) \cap \big(F_2, S(F_2)\big) = \big(F_1 \cap F_2, S(F_1) \cup S(F_2)\big) \quad (3)$$

$$\big(F_1, S(F_1)\big) \cup \big(F_2, S(F_2)\big) = \big(F_1 \cup F_2, S(F_1) \cap S(F_2)\big) \quad (4)$$

We note $F_1 \cap F_2$, $F_1 \cup F_2$ for short. From now on we consider a functional deficiency to be complete when not explicitly mentioned otherwise. Also, we sometimes write a functional deficiency as the conjunction of its elements. The tuple $\big(F, S(F)\big)$ is denoted a *reconfiguration tuple*. Finally, it is possible to prioritize[6] a functional deficiency $pr(F) = \sum_{i=1}^{n} \sum_{j=1}^{f} p(s_N^i)p(s_F^j)$, $(s_N^i, s_F^j) \in S(F)$.

Definition 3 (Core functional deficiency). *The core functional deficiency F^c has its elements satisfied in* all *states of $\mathcal{P}(A)$ and* deficient *in* all *states of $\mathcal{D}(A)$. F^c is unique for a given set $\mathcal{S}$, and its priority is equal to* 1.[7]

Note that at least all misbehaving variables in states of $S(F)$ do belong to the core deficiency, as does $Q = 0$ in our example.

3.3 Minimal functionalities over maximal reconfiguration sets

This section develops a characterization of functional deficiencies whose size is minimal, while deficient over the largest number of state estimates. From properties 1 and 2, the reconfiguration set increases in size when the functionality decreases in size. A complete functional deficiency of minimal size over a maximal reconfiguration set is then easily characterized.

Definition 4 (Minimal functional deficiency over the maximal reconfiguration set). *A minimal functional deficiency F has a maximal reconfiguration set $S(F)$ if it exists no other functional deficiency F' such that $S(F) \subset S(F')$ and $F' \subset F$.*

The search for *minimal* functional deficiencies over *maximal* reconfiguration sets leads to a set of functional deficiencies denoted *minimax*. A minimax functional deficiency represents the minimum set of variable instances that are deficient over the same maximum set of pairs of nominal/faulty states.

Proposition 1. *Given two minimax functional deficiencies F and F' such that $F' \cap F \neq \emptyset$, then $S(F') = S(F)$.*

Proof. If $F'' = F' \cap F$ and $F'' \neq \emptyset$, then if $F'' \subset F$, from definition 4 and property 1, it comes $S(F) = S(F'')$. Similarly, $F'' \subset F'$ yields $S(F'') = S(F')$, so $S(F) = S(F')$. The same result is obtained if $F'' = F$ or $F'' = F'$ with property 2. □

The previous proposition implicitly focuses the search on *distinct minimax functionalities*. Thus functional deficiencies may be characterized as disjoint sets of variable instances. This result brings flexibility to the reconfiguration process under uncertainty, but is mitigated as the disjoint functions are not independent from each other w.r.t. to the hybrid dynamics. In other words, they may not be recovered independently. In reference to the recovery (planning) operation, these functionalities are no serializable goals.

Proposition 2. *The core functional deficiency F^c is minimax.*

Proof. This is trivial from definition 4. F^c is also complete with $S(F^c) = \mathcal{S}$. □

[6] Note that in this expression, there is no notion of fault criticality. Every faulty state is assumed to have equal criticality but the probability of the state is taken into account.

[7] Given that $\mathcal{P}(A)$ and $\mathcal{D}(\mathcal{A})$ have their state probabilities summing to 1.

3.4 Functional Deficiencies Computation

We note that: first, F^c is minimax and easily computed, second, minimax deficiencies are non-intersecting, third, deficiencies over small reconfiguration sets are larger than those over large reconfiguration sets. Therefore a way to compute the minimax deficiency is to strip rough non-minimax deficiencies over trivial small reconfiguration sets from their intersection with F^c. Similarly, intersections among non-minimax deficiencies form new sets (as they cannot intersect other sets). Algorithm 1 progressively reduces simple complete, but non minimax deficiencies. Its first step updates the deficiencies for each combination of two states of $\mathcal{S}$ using the measure of relation (1), and computes the core function. Iterating through this set, step 3 prunes out a deficiency of its intersection with F^c. Step 4 prunes out intersecting deficiencies of their intersection and creates a new deficiency from it. Step 5 merges the reconfiguration sets of similar deficiencies. The algorithm is better understood by developing our example. Step 1 gives:

```
1: Compute the complete $F$ w.r.t. each reconfiguration set $(s_N^p, s_F^q)$, compute $F^c$, and add them all to the agenda.
2: Iterate through the tuples $(F_i, F_j)$ in the agenda.
3: If $F^c \cap F_i \neq \emptyset$, $F_i \longleftarrow F_i \setminus \{F_i \cap F^c\}$.
4: Else if $F_i \cap F_j \neq \emptyset$, create a new function $F' = F_i \cap F_j$ and add it to the agenda. Do $F_i \longleftarrow F_i \setminus F'$.
5: Else if $F_i = F_j$, $S(F_i) = S(F_i) \cup S(F_j)$ and remove the remaining function $F_j$ from the agenda.
6: $F_i$ is minimax when it does not intersect with other functions anymore. It is removed to the agenda and returned.
```

Algorithm 1: Computing minimax functional deficiencies

$$
\begin{array}{rcl}
s_N^1, s_F^1 & : & F_1 = (V_1.m = \mathit{open}) \wedge Q_1 > 0 \wedge Q > 0 \\
s_N^1, s_F^2 & : & F_2 = P_0 < P^* \wedge (S.m = \mathit{open}) \\
 & & \wedge (V_2.m = \mathit{closed}) \wedge Q_1 > 0 \wedge Q > 0 \\
 & & \wedge (V_1.m = \mathit{open}) \\
s_N^1, s_F^3 & : & F_3 = P_0 < P^* \wedge (S.m = \mathit{open}) \\
 & & \wedge Q_1 > 0 \wedge Q > 0 \wedge (V_1.m = \mathit{open}) \\
s_N^2, s_F^1 & : & F_4 = P_0 \geq P^* \wedge (S.m = \mathit{closed}) \\
 & & \wedge (V_1.m = \mathit{open}) \wedge (V_2.m = \mathit{open}) \\
 & & \wedge Q_1 > 0 \wedge Q_2 > 0 \wedge Q > 0 \\
s_N^2, s_F^2 & : & F_5 = (V_1.m = \mathit{open}) \wedge (V_2.m = \mathit{open}) \\
 & & \wedge Q_1 > 0 \wedge Q_2 > 0 \wedge Q > 0 \\
s_N^2, s_F^3 & : & F_6 = (S.m = \mathit{closed}) \wedge (V_1.m = \mathit{open}) \\
 & & \wedge Q_1 > 0 \wedge Q_2 > 0 \wedge Q > 0 \\
 & & \wedge (V_2.m = \mathit{open}) \\
s_N^1, s_N^2, s_F^1, s_F^2, s_F^3 & : & F^c = (V_1.m = \mathit{open}) \wedge Q_1 > 0 \wedge Q > 0
\end{array}
$$

We have $F_1 = F^c$ so F_1 can be eliminated. Then reducing other functions with F^c:

$$
\begin{array}{rcl}
F_2 & = & P_0 < P^* \wedge (S.m = \mathit{open}) \wedge (V_2.m = \mathit{closed}) \\
F_3 & = & P_0 < P^* \wedge (S.m = \mathit{open}) \\
F_4 & = & P_0 \geq P^* \wedge (S.m = \mathit{closed}) \wedge (V_2.m = \mathit{open}) \wedge Q_2 > 0 \\
F_5 & = & (V_2.m = \mathit{open}) \wedge Q_2 > 0 \\
F_6 & = & (S.m = \mathit{closed}) \wedge Q_2 > 0 \wedge (V_2.m = \mathit{open})
\end{array}
$$

1. $F_2 \cap F_3 = P_0 < P^* \wedge (S.m = \mathit{open})$, $F_7 \longleftarrow P_0 < P^* \wedge (S.m = \mathit{open})$, $S(F_7) = (s_N^1; s_F^2, s_F^3)$, $F_2 = F_2 \setminus F_7 = (V_2.m = \mathit{closed})$, $S(F_2) = (s_N^1; s_F^2)$. F_7 is added to the agenda.
2. $F_2 \cap F_4 = \emptyset$, $F_2 \cap F_5 = \emptyset$, $F_2 \cap F_6 = \emptyset$, and $F_2 = V_2.m = \mathit{closed}$ is minimax.
3. $F_3 \cap F_4 = \emptyset$, $F_3 \cap F_5 = \emptyset$, $F_3 \cap F_6 = \emptyset$, $F_3 = F_7$, remove F_7, $S(F_3) = (s_N^1; s_F^2, s_F^3)$. $F_3 = P_0 < P^* \wedge (S.m = \mathit{open})$ is minimax.
4. $F_4 \cap F_5 = F_5$, $F_4 \longleftarrow F_4 \setminus F_5 = P_0 \geq P^* \wedge (S.m = \mathit{closed})$, $S(F_4) = (s_N^2; s_F^1)$. $S(F^5) = (s_N^2; s_F^1, s_F^2)$.
5. $F_4 \cap F_6 = (S.m = \mathit{closed})$, $F_8 = (S.m = \mathit{closed})$, $S(F_8) = (s_N^2; s_F^1, s_F^3)$, $F_4 \longleftarrow F_4 \setminus F_8 = P_0 \geq P^*$, $S(F_4) = (s_N^2; s_F^1)$, and F_4 is minimax.
6. $F_6 \cap F_5 = F_5$, $F_6 \longleftarrow F_6 \setminus F_5 = F_8$. Remove F_8, $F_6 = (S.m = \mathit{closed})$, $S(F_6) = (s_N^2; s_F^1, s_F^3)$. F_5, F_6 are minimax. $S(F^5) = (s_N^2; s_F^1, s_F^2, s_F^3)$.

Finally, the minimax functions are:

$$
\begin{array}{l}
F^c = (V_1.m = \mathit{open}) \wedge Q_1 > 0 \wedge Q > 0 \, , S(F^c) = (s_N^1, s_N^2; s_F^1, s_F^2, s_F^3) \\
F_2 = (V_2.m = \mathit{closed}) \, , S(F_2) = (s_N^1; s_F^2) \\
F_3 = P_0 < P^* \wedge (S.m = \mathit{open}) \, , S(F_3) = (s_N^1; s_F^2, s_F^3) \\
F_4 = P_0 \geq P^* \, , S(F_4) = (s_N^2; s_F^1) \\
F_5 = (V_2.m = \mathit{open}) \wedge Q_2 > 0 \, , S(F_5) = (s_N^2; s_F^1, s_F^2, s_F^3) \\
F_6 = (S.m = \mathit{closed}) \, , S(F_6) = (s_N^2; s_F^1, s_F^3)
\end{array}
$$

We distinguish the *continuous reduction* of F_i, that is its reduction to variables in $\mathcal{X}$, from the *hybrid* deficiency (made of both discrete and continuous instances). Intuitively, as the modes are relaxed, there exist more states that satisfy the continuous reduction to a deficiency, than the hybrid deficiency. For this reason, we say the latter leads to *reset* solutions (as modes deficiencies are explicitly set up to be recovered), as opposed to *redundancy* solutions (modes are unspecified, several component modes may recover the continuous deficiencies). We note $\bar{F}$ the continuous reduction to F.

4 Reconfiguration of Functional Deficiencies

Previous works in $MBReconf$ have shown how to deduce goal states from discrete configuration goals generated by a high level planner [16] and to recover them [17]. These techniques do not scale well to systems with continuous state and behavior: first, the space to be explored is no more finite; second, no bijective mapping exists between continuous variable instances and a mode. Reachability analysis [7] searches for the set of hybrid states that can be reached from a set of initial conditions. Extensions include techniques for the automatic design of controllers under safety conditions [1]. However, these techniques are still computationally expensive. Here, we propose to determine the goal configurations through a process similar to the model-based diagnosis consistency approach. Indeed, reconfiguration can be viewed as the problem of identifying components whose reconfiguration is sufficient to restore acceptable behavior, when diagnosis is the problem of identifying component modes whose abnormality is sufficient to explain observed malfunctions [5]. Besides modes, we search for the sufficient conditions over the continuous space that are required to recover a given deficiency. Such conditions delimitate behavioral regions that we refer to as configurations of the hybrid system. This strategy requires a static representation of the behavioral equations and a set of algorithms to reason about it.

In the following, we denote as the *goal functional deficiency* F^* the deficiency to be recovered. Its selection is part of the recovery process, and is detailed at the end of this section. For now, we pick up a simple F^* as F^c because its priority is maximal, and it covers all state estimates.

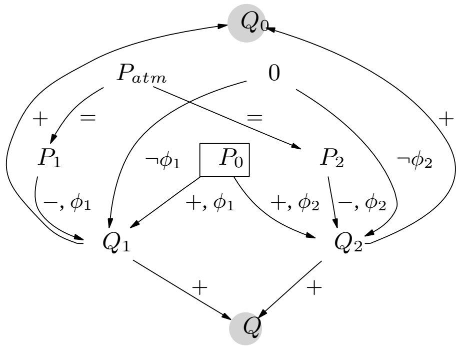


Figure 2. Pressure expansion system causal-graph

4.1 Configurations identification

4.1.1 Causal-graph of influences

Reachability analysis is a time-analysis; by applying model-based reasoning techniques we collapse this analysis at a single point in time. Therefore a static representation of the equations in E is needed. We use a causal representation.

Definition 5 (Causal-Graph of Influences). *The causal-graph of influences of a set of equations E is an oriented graph $G = (X, I)$ where the variables in X form a set of nodes x_i, and I a set of arcs among these variables.*

The causal-graph is a representation of relations among variables in E that holds at every time step.

Definition 6 (Causal Influence). *A causal influence in I, $I_{i,j} = (x_i, x_j, b, \phi)$, is a directed arc between two variables x_i and x_j, with b the* sign *of the influence and ϕ its* activation condition.

Influences are drawn from the implicit causality in E. Inputs are subject to no influence. Figure 2 depicts the causal-graph of the pressure expansion system. In the following we replace equations in E with G. Note that in general some work is required to extract the causality from static relations [15]. $b = \{-1, 1\}$ stores the numerical *positive or (equal or negative)* influence among variables. ϕ's truth value in the hybrid state determines the *activation*/*deactivation* of the influence in the graph. Unconditioned, the influence is permanently active.

Definition 7 (Configuration). *A configuration for G (and by extension A) is of the form $\bigwedge_i \phi_i$.*

A configuration corresponds to a certain combination of activated influences in the graph, and delimits a region of behavior of A. In our example, $V_1.m = open \wedge V_2.m = open \wedge P_0 \geq P^* \wedge P_0 \geq P_1 \wedge P_0 \geq P_2 \wedge S.m = closed$ is a nominal configuration of the system.

4.1.2 Building configuration goals from functional deficiencies

We write the MBD theory based on consistency [13] where for the reconfiguration purpose, observations are replaced with functional deficiencies. A deficiency F^* has been characterized w.r.t. the state uncertainty. We are now searching for the *minimal sets of conditions* that are sufficient to restore F^*.

Definition 8 (Reconfiguration candidate). *A reconfiguration candidate for A given F^* is defined as a minimal set $\{I_1^\Delta, \cdots, I_h^\Delta\} \subseteq I$ of influences such that*

$$A \cup F^* \cup \neg\phi_1^\Delta \cup \cdots \cup \neg\phi_h^\Delta \qquad (5)$$

is consistent. We note a condidate $\Delta = \{\phi_1^\Delta, \cdots, \phi_h^\Delta\}$.

Definition 9 (Reconfiguration conflict). *A reconfiguration conflict for A given F^* is a set $\lambda = \{I_1^c, \cdots, I_k^c\}$ of influences such that*

$$A \cup F^* \cup \phi_1^c \cup \cdots \cup \phi_k^c \qquad (6)$$

is not consistent.

From $G \cup F^*$ and assuming a faulty configuration (a configuration consistent with some state in $S_F(F^*)$), we seek for reconfiguration conflicts in G that are such that influences in a conflict cannot be activated together given F^*. For a deficient variable (node) x_j of F^*, we call *ascending* influences the influences that belong to the paths from the inputs/other deficient variables, to x_j. An ascending influence I_i for x_j is noted $\lambda_i^j = \{I_i, \phi_i\}$. A conflict for x_j is thus the set λ^j of its ascending influences $\{\lambda_i^j\}_{i=1,\cdots,n_j}$. $\Lambda = \left\{\{\lambda^j\}_{j=1,\cdots,n_{F^*}}\right\}$ is the collection of conflicts over all deficient variables of F^*. The set of minimal sets of influences that are responsible for the conflicts is obtained similarly to the minimal diagnoses in the MBD theory by computing the hitting sets (HS) over Λ [13]. Following definition (5) we note $\Delta_q = (\wedge_{I_i \in \mathcal{I}_q} \phi_i)$ where $\mathcal{I}_q$ is a set of influences that must all be deactivated for restoring F^*. Consequently, we note $\Delta = \left\{\{\Delta_q\}_{q=1,\cdots,n_q}\right\}$. Sufficient conditions in Δ_q are those such that influences in $\mathcal{I}_q$ are all deactivated. Therefore, they are given by $\neg\Delta_q$, and we note $\neg\Delta = \left\{\{\neg\Delta_q\}_{q=1,\cdots,n_q}\right\}$. The associated goal configurations are given by $\neg\Delta_q \wedge F^*$. A goal configuration characterizes a set of goal states in which variables that are not specified in its definition can take any value.

```
1: Apply $F^*$ to $G$ (remove the deficiency and assume conditions
   corresponding to applied variable instances to be true).
2: Apply a faulty configuration consistent with $S_F(F^*)$ to $G \setminus F^*$.
3: Get the conflicts $\Lambda$.
4: Compute $\Delta = HS(\Lambda)$.
5: $\neg\Delta \wedge F^*$ are goal configurations.
```

Algorithm 2: Identifying goal reconfiguration candidates ($Goals$)

Consider our example again. Reconfiguring $F^* = F^c$ with algorithm 2 implies ϕ_1 is satisfied (step 1), and based on remaining variable instances in states in $S_F(F^*)$ the configuration of the subgraph $G \setminus F^*$ (G deprived of nodes and axis to nodes in F^*) is determined, in that case $\neg\phi_2$ is satisfied (V_2 is closed or stuck closed in all faulty states) (step 2)[8]. Tracing the ascending influences in G, it comes two sets of conflicts (one per continuous variable instance in F^*):

$$\begin{cases} \lambda_Q = \{Q \leftarrow Q_1, Q \leftarrow Q_2, Q_2 \overset{\neg\phi_2}{\leftarrow} 0, P_2 \leftarrow P_{atm}\} \\ \lambda_{Q_1} = \{Q_1 \overset{\phi_1}{\leftarrow} P_0, Q_1 \overset{\phi_1}{\leftarrow} P_1, P_1 \leftarrow P_{atm}\} \end{cases}$$

ϕ_1 is satisfied in F^c, and influences over Q, P_1 and P_2 are activated in all configurations, so it simplifies to:

$$\begin{cases} \lambda_Q = \{Q_2 \overset{\neg\phi_2}{\leftarrow} 0\} \\ \lambda_{Q_1} = \{\} \end{cases}, \Lambda = \{\lambda_Q, \lambda_{Q_1}\}$$

[8] Some discrete values and conditions may be undetermined, then several initial configurations must be considered. We take the probabilized mean for continuous values.

It comes $\Delta = \{\{\neg\phi_2\}\}$ and $\phi_2 \wedge F^c$ is a goal configuration (step 5). Reconfiguring the continuous reduction $\bar{F}^c$ leads to more opportunities: ϕ_1 is no more satisfied and $\lambda_{Q_1} = \{Q_1 \overset{\neg\phi_1}{\leftarrow}\}$, thus $\Delta = \{\{\neg\phi_1, \neg\phi_2\}\}$ and configuration goals are given by $\phi_1 \wedge \phi_2 \wedge \bar{F}^c$.

4.2 Recovery

In this subsection we sketch our future strategy for the recovery of the configuration goals. The recovery operation aims at bringing the system into the regions defined by the configuration goals. Due to the hybrid dynamics, a solution is a chain of transitions to the component mode goals, while the continuous dynamics ensure the transition guards are successively satisfied.

The search for the right succession of transitions defines a probabilistic conformant planning problem [10], where a set of transitions must bring the system to a set of predetermined goals, under uncertainty and without observing the system full state. The plan maximizes the probability of reaching the goal configuration given the initial belief state $\mathcal{D}(A)$. In our example, a stuck valve cannot be reopened, so no plan exists to restore deficiencies F^c and $\bar{F}^c$. A plan exists to restore F_5 for some of the initial states, $Pl = \{\tau_3, \tau_{21}\}$. F_6 has a plan $Pl = \{\tau_3\}$.

Satisfying successive transition guards ϕ_j through system inputs defines a control problem. A model predictive control (MPC) approach solves on-line a finite horizon open-loop optimal control problem subject to system dynamics and constraints involving states and controls. Based on measurements obtained at time k, the future dynamic behavior of the system is predicted over a fixed horizon, and the controller determines the input such that a performance criterion is optimized. This technique fits well within the model-based autonomous system framework, given that two key elements are already present, the model A, and the state predictor (or estimator) $\mathcal{P}(A)$.

Solving this control problem for complex system however requires more research. The MPC community itself seeks for better integration of modern state estimation techniques within the control loop [11]. We are currently working on an interleaved planning and control scheme as an extension to existing optimal control methods for hybrid systems [4].

Our general strategy to the reconfiguration of the functional deficiencies explores *reset* solutions first, then *redundancy* solutions (continuous reductions) in prioritized order. In case of plan failure the next deficiency is selected (see algorithm 3). In our example, s_F^2 and s_F^3 have much lower probability than s_F^1 as they correspond to multiple faults. F^c is subject to plan failure. F_6: $S.m = closed$ is its own goal configuration and has a plan τ_3 whose guard is $P_0 \geq P^*$. MPC generates the pressure input P_0 to reach P^*. Note that depending on the real initial state, the reconfiguration may have no effect. The operation does not harm the system as we consider that maintaining a nominal level of pressure does not harm even the faulty system. Besides, it helps to discriminate among the estimates: for example, if reconfiguring F_6 fails, s_F^1, and potentially s_F^2 are eliminated.

```
1: Compute functional deficiencies with algorithm 1
2: Identify goal configurations with algorithm 2.
3: Find a plan, in case of failure move to the next deficiency, in
   prioritized order.
4: Apply MPC using $\mathcal{P}(A)$ as the predictor.
```

Algorithm 3: Prioritized selection of functional deficiencies

5 Summary, Existing works and Perspectives

We have presented a methodology for the automated reconfiguration of functional deficiencies. The deficiencies are identified by comparing predicted and diagnosed states, and then partitioned and prioritized w.r.t. the belief over the state estimates. Goals are further identified from the deficiencies without proceeding to a reachability analysis.

A pioneer work, [5], explores the analogy between the problems of diagnosis and reconfiguration. Goal identification and safe planning have been studied in [17] in the case of qualitative models. We are not aware of any work on the planning of hybrid systems.

Finally, it appears that restoring a single minimax deficiency does not restore a fully nominal state: an alternate strategy would be to combine the deficiencies so to restore a single nominal state instead. In a near future, we will better experiment the recovery process and study its relation to hybrid system control.

REFERENCES

[1] E. Asarin, O. Bournez, T. Dang, O. Maler, and A. Pnueli, ‘Effective controller synthesis of switching controllers for linear systems’, *Proceedings of the IEEE, Special Issue on Hybrid Systems*, **88**, 1011–1025, (July 2001).

[2] E. Benazera and L. Travé-Massuyès, ‘The consistency approach to the on-line prediction of hybrid system configurations’, in *Proceedings of the IFAC Conference on Analysis and Design of Hybrid Systems 2003*, (2003).

[3] O. Bournez, *Complexit algorithmique des systmes dynamiques continus et hybrides*, Ph.D. dissertation, Ecole Normale Suprieure de Lyon, 1999.

[4] C.G. Cassandras, D.L. Pepyne, and Y. Wardi, ‘Optimal control of a class of hybrid systems’, *IEEE Trans. on Automatic Control*, **46**(3), 398–415, (2001).

[5] J. Crow and J. Rushby, ‘Model-based reconfiguration: toward an integration with diagnosis’, in *Proceedings of AAAI-91, Anaheim, CA*, volume 2, pp. 836–841, (1991).

[6] W. Hamscher, L. Console, and J. De Kleer, *Readings in Model-Based Diagnosis*, Morgan Kaufmann, San Mateo, CA, 1992.

[7] T. A. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi, ‘Beyond HYTECH: Hybrid systems analysis using interval numerical methods’, in *HSCC*, pp. 130–144, (2000).

[8] M. Hofbaur and B.C. Williams, ‘Mode estimation of probabilistic hybrid systems’, *Hybrid Systems: Computation and Control, Lecture Notes in Computer Science (HSCC 2002)*, **2289**, 253–266, (2002).

[9] F. Hutter and R. Dearden, ‘The gaussian particle filter for diagnosis of non-linear systems’, in *Proceedings of the Thirteenth International Workshop on Principles of Diagnosis DX-03*, (2003).

[10] N. Hyafil and F. Bacchus, ‘Conformant probabilistic planning via csps’, in *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS 03)*, (2003).

[11] M. Morari and J. H. Lee, ‘Model predictive control: past, present, future’, in *Joint 6th International Symposium on Process Systems Engineering (PSE’97)*, (1997).

[12] P. Nayak and J. Kurien, ‘Back to the future for consistency-based trajectory tracking’, in *Proceedings of AAAI-2000, Austin, Texas*, (2000).

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

[14] M. Stumptner and F. Wotawa, ‘Reconfiguration using model-based diagnosis’, in *Proceedings of the Tenth International Workshop on Principles of Diagnosis DX-99*, (1999).

[15] L. Travé-Massuyès and R. Pons, ‘Causal ordering for multiple modes systems’, in *Proceedings of the Eleventh International Workshop on Qualitative Reasoning*, pp. 203 – 214, (1997).

[16] B. C. Williams and P. Nayak, ‘A model-based approach to reactive self-configuring systems’, in *Proceedings of AAAI-96, Portland, Oregon*, pp. 971–978, (1996).

[17] B. C. Williams and P. Nayak, ‘A reactive planner for a model-based executive’, in *Proceedings of IJCAI-97*, (1997).