Fault Isolability Prediction of Diagnostic Models

Mattias Krysander & Mattias Nyberg
Linköpings universitet
Department of Electrical Engineering
SE - 581 83 Linköping, Sweden
{matkr, matny}@isy.liu.se

Abstract

Fault isolability plays a significant role and could be critical with respect to many aspects such as safety and maintenance for a process to be diagnosed. In the development of processes including diagnosis, design decisions are taken, e.g. sensor configuration selection, which affects the fault isolability possibilities. In this paper an algorithm for predicting fault isolability possibilities using a structural model describing the process is proposed. Since only a structural model is needed as input, the algorithm can easily predict fault isolability possibilities of different design concepts. In contrast to previous algorithms using structural models no assumption is imposed on the model. The algorithm computes faults that cannot be distinguished from other faults, which can be used to exclude design alternatives with insufficient isolability possibility.

1 Introduction

Fault isolability refers to the question of which faults that are possible to distinguish from other faults, given the knowledge of available sensor and actuator signals. This information is important when designing diagnostic systems but also when designing the process to be diagnosed.

In the development of processes, different design decisions are taken, e.g. how different parts are connected, which actuators to use, and which sensors to use. All these design decisions may influence the isolability possibilities. In addition, when designing the diagnostic system, there is a choice of different fault modeling strategies and which diagnostic tests to include. As a guidance when taking these design decisions, it is desirable to know exactly how different design choices affect the isolability possibilities.

To find the isolability of a given model of a process is a difficult problem in general since it is related to the problem of solving large systems of non-linear differential equations. In this paper we attack the problem by an algorithm that takes a structural model of a process as input and computes faults that are not isolable from other faults. Since only a structural model is used, no precise analytical equations are needed. This implies that the algorithm can be used early in the design phase and thus serve as a guidance when taking different design decisions. However, if we need to know exactly which faults that are isolable from others, the algorithm also helps braking down the large problem into smaller and easier problems to analyze.

Isolability analysis has previously been studied in [O. Dressler, 2003], but only for qualitative models. Furthermore, a structural method for computing the isolability of different sensor configurations was presented in [Travé-Massuyès *et al.*, 2003]. This and other earlier works using structural models for diagnosis, e.g. [Pulido and Alonso, 2002], [Frisk *et al.*, 2003], [Cassar and Staroswiecki, 1997], and [Blanke *et al.*, 2003], have imposed analytical assumptions on the systems, e.g. that only subsystem with more equations than unknowns, i.e. only over-constrained subsystems, can be invalidated and therefore contribute to detection and isolation. However these assumptions are difficult to verify in most larger models. If these assumptions are not satisfied, faults that are predicted to be isolable from other faults can be not isolable and vice verse. In contrast, the method presented in this paper does not require any analytical assumptions.

In Section 2 a modeling framework for model based diagnosis is recapitulated. In Section 3 the central concepts detectability and isolability are recalled. These concepts are related to structural properties of the model through the new concept of *checking model* presented in Section 4. We describe how checking models can be computed by using a structural model. By combining the algorithm for finding checking models with the results relating checking models and isolability, an algorithm for isolability prediction is developed in Section 5. An example shows how the obtained isolability prediction can be interpreted. Furthermore, in Section 6 illustrative examples show how isolability prediction can be used to identify additional fault modeling and support sensor selection to meet given isolability requirements.

2 Example Introduction and Models

Throughout the paper, we will exemplify concepts and techniques on the same example, i.e. the water-tank process depicted in Figure 1. The water-tank process consists of a pump, a tank, a water-level sensor, and a flow sensor. These components are denoted P, T, W, and Q respectively and are illustrated in the figure by the four dashed boxes. The pump is pumping water into the top of the tank. The water flows out of the tank through a pipe connected to the bottom of the tank. The pump is controlled by a control signal u, the water-level in the tank is measured with the sensor signal y_w, and the outflow from the tank is measured with the sensor signal y_q. The true flows into and out of the tank are denoted q_i, and the actual water level in the tank is denoted w.

A physical model of the process is shown in Table 1. The model is organized according to the modeling principles given in [Dressler *et al.*, 1993; Nyberg and Krysander,

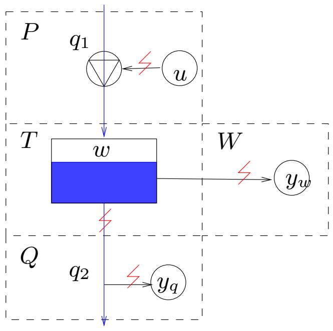

Figure 1: The process to be diagnosed. The location of possible faults are denoted with a red flash.

Table 1: A model for the water-tank process in Figure 1.

Assumption	Equation	Expression
Pump		
$P = \text{NF}$	e_1	$u = q_1$
Tank		
	e_2	$w' = q_1 - q_2$
	e_3	$w = (1 - f_t)q_2^2$
	e_4	$w' = 2\,(1 - f_t)\,q_2 q_2' - f_t'\,q_2^2$
	e_5	$f_t' = 0$
$T = \text{NF}$	e_6	$f_t = 0$
Water-level sensor		
$W = \text{NF}$	e_7	$y_w = w$
$W = \text{NF}$	e_8	$\dot{y}_w = w'$
Flow sensor		
	e_9	$y_q = q_2 + f_q$
	e_{10}	$\dot{y}_q = q_2' + f_q'$
	e_{11}	$f_q' = 0$
$Q = \text{NF}$	e_{12}	$f_q = 0$

2003]. The equation e_1 describes the pump; e_2 the conservation of volume in the tank; e_3 and e_4 the outflow from the tank caused by the gravity and with a possible clogging fault f_t; e_5 a fault model for the clogging fault; e_6 the no-fault value for fault variable f_t; e_7 and e_8 the fault free water-level measurement; e_9 and e_{10} the outflow measurement with a possible bias fault f_q; and e_{11} and e_{12} the outflow-measurement fault f_q. Note that both arbitrary faults, e.g. the water-level sensor fault, and faults modeled by fault parameters, e.g. the bias fault of the outflow measurement, can be handled by this modeling principle.

By including analytically differentiated equations, i.e. e_4, e_8, and e_{10} in the example, the derivatives of the unknowns can be replaced with new algebraic variables. Thus a derivative $\dot{x}$ is eliminated by substituting a so called *dummy derivative* x' [Mattson and Söderlind, 1993] for $\dot{x}$ wherever it occurs in the model. Although we assume that $x' = \dot{x}$, this is not true by definition, instead this relationship should be implied by the augmented algebraic model containing differentiated equations. For example w' is an algebraic variable, i.e. it is not defined as the derivative $\dot{w}$ of w, but should be equal to $\dot{w}$. The algebraic equations e_7 and e_8 together with the differential equation $\dot{y}_w = dy_w/dt$ imply that $w' = dw/dt = \dot{w}$. In this way it is possible to transform an over-constrained system of differential-algebraic equations into an algebraic system. The price paid for converting a differential algebraic model into an algebraic model is that the number of equations grows. The conversion from a differential algebraic model to an algebraic model can be done using an algorithm in [Krysander and Nyberg, 2002].

The assumption of the first equation, i.e. $P = \text{NF}$, means that $u = q_1$ is valid if the *behavioral mode* of component P is in the no-fault mode, which is abbreviated NF. For the water-tank example all components are assumed to be either in no-fault mode NF or in faulty mode F. Equations with no assumptions are always true. A mode assignment for all components of a process is called a *system behavioral-mode*. The no-fault system behavioral-mode for the water-tank process will be denoted $\mathbf{NF}$ and fault modes will be denoted by their faulty components, e.g. $\mathbf{PT}$ for the behavioral mode where components P and T are in faulty mode and W and Q are in no-fault mode.

The set of equations that are valid in a given system behavioral-mode b, i.e. its *behavioral model* denoted M_b, defines the behavior of process in system behavioral-mode b. For an example, the set of all equations except e_7 and e_8 is the behavioral model of behavioral-mode $\mathbf{W}$.

3 Detectability and Isolability Prediction

First some definitions are briefly introduced. An *observation* is here considered to be a snap-shot of all known variables and possibly also some derivatives of known variables. For the water-tank process an observation is a value of the vector $[u(t), y_w(t), \dot{y}_w(t), y_q(t), \dot{y}_q(t)]$ at time t. A *diagnosis* at time t is a system behavioral-mode such that its behavioral model is consistent with the observation at time t. A system behavioral-mode b_i is said to be *isolable* from another system behavioral-mode b_j if there exists some observation such that b_i is a diagnosis but b_j is not. A fault b_i is said to be *detectable* if it is isolable from the no-fault system behavioral mode.

It could be argued that the proposed definition of detectability is relatively weak in the sense that a fault is detectable if there exists only one single observation that distinguish the fault from the no-fault mode. However, by using this relatively weak definition, a non-detectable fault would also be non-detectable with any stronger definition of detectability.

3.1 Predicting Detectability

In this section we will describe how detectability information can be derived without knowing the exact analytical equations of a model like the one in Table 1. It can be realized that b is not detectable if $M_{\mathbf{NF}} \subseteq M_b$. However detectability analysis by this naive idea comparing behavioral models is not particularly powerful. Here a refinement of this idea will be presented.

Consider first the no-fault system behavioral-model. As in [Blanke *et al.*, 2003], a fault can *violate* some equations in the no-fault system-behavioral model, i.e. some equations in no-fault system-behavioral model can be false for variable values consistent with the behavioral model of the fault. For example the fault of the outflow sensor $\mathbf{Q}$ in the water-tank example can violate e_{12} in the no-fault system behavioral-model $M_{\mathbf{NF}}$.

Even if a fault can violate an equation in a model, it is not sure that the fault is detectable as the next small illustrative example shows. Consider a no-fault behavioral model $M_{\mathbf{NF}}$ defined as

$$u = x_1 \tag{1a}$$
$$y = x_1 \tag{1b}$$
$$0 = x_1 + x_2 \tag{1c}$$

where u and y are known variables and x_1 and x_2 are unknowns. The set of observations consistent with (1a)-(1c),

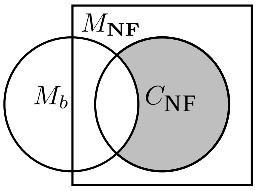

Figure 2: Venn-diagram representation of equation sets.

i.e. $M_{\mathbf{NF}}$ is

$$\{(u, y) \in \mathbb{R}^2 | u = y\} \quad (2)$$

which will be called the *observation set* for $M_{\mathbf{NF}}$ and denoted $\mathcal{O}(M_{\mathbf{NF}})$. A fault violating either (1a) or (1b) is detectable, because $u \neq y$ if either (1a) or (1b) is violated, i.e. (u, y) belongs not to the observation set (2). A fault which only violates (1c) cannot be detected because a violation of (1c) leads to different values of x_2 but $u = y$ still holds. Equation (1c) is therefore said to be *non-monitorable* in [Blanke *et al.*, 2003].

A difference between the first two equations where a fault can be detected and (1c) where a fault can not be detected is that the first two equations define the observation set (2) and (1c) is not needed to define (2). Observation set is next defined to formalize this discussion. If M is a set of equations, $\mathbf{x}$ a vector of unknowns, and $\mathbf{z}$ a vector of known variables, then the observation set for M is defined by $\mathcal{O}(M) = \{\mathbf{z} | \exists \mathbf{x} \wedge_{e \in M} e(\mathbf{x}, \mathbf{z})\}$. The following definition will be used to formalize in which equations violations can be detected.

Definition 1 (C_b, Checking Model of b) *A model C_b is a* checking model of b *if C_b is a subset of the behavioral model M_b and $\mathcal{O}(C_b) = \mathcal{O}(M_b)$.*

Note that behavioral models trivially are checking models. Note also that checking models need not be over-constrained. As examples of checking models, the two checking models of $\mathbf{NF}$ in (1) are the sets {(1a), (1b)} and {(1a),(1b), (1c)}. A detectable fault violates at least one equation in every checking model $C_{\mathbf{NF}}$ for the no-fault behavioral mode. A detectable fault must therefore violate (1a) or (1b) in (1), because {(1a), (1b)} is a checking model of $\mathbf{NF}$.

An illustration of the equation sets involved in the discussion is shown in Figure 2 as a Venn diagram. The rectangle represents the set of all equations in the no-fault behavioral model $M_{\mathbf{NF}}$, i.e. (1a)-(1c) in the small example. The right circle contains a checking model C_{NF} of the no-fault behavioral mode, i.e. (1a)-(1b) in the example. The left circle contains the behavioral model M_b for some behavioral mode b. The grey-shaded area represents the set of equations which can be violated in behavioral mode b, i.e. the equations that render detection of behavioral mode b possible. Hence if the grey-shaded area is empty, then b is not detectable. If M_b = {(1a),(1b)} in the example with $M_{\mathbf{NF}}$ equal to (1) then b is not detectable, because both (1a) and (1b) hold in b. From this discussion the next theorem follows which summarizes how checking models will be used for detectability analysis.

Theorem 1 *A system behavioral-mode b is not detectable if there exists a checking model $C_{\mathbf{NF}}$ of $\mathbf{NF}$ such that $C_{\mathbf{NF}} \subseteq M_b$.*

The proof of Theorem 1 and the proofs of all following theorems can be found in [Krysander and Nyberg, 2005]. How to find checking models will be described in Section 4.

3.2 Predicting Isolability

Since detectability is a special case of isolability, the results of Theorem 1 concerning detectability can be generalized to isolability as follows. A behavioral mode b_i, that is isolable from a behavioral mode b_j, violates some equations in a checking model C_{b_j} of the behavioral mode b_j. Figure 2 could represent this situation as well if $\mathbf{NF}$ is changed to b_j and b to b_i. Then it can be seen that if all equations in a checking model C_{b_j} hold in behavioral mode b_i then it follows that b_i is not isolable from b_j. Hence by computing a checking model of C_{b_j}, it can be concluded which behavioral modes that are not isolable from b_j.

Theorem 2 *A system behavioral-mode b_i is not isolable from a system behavioral mode b_j if there exists a checking model C_{b_j} of b_j such that*

$$C_{b_j} \subseteq M_{b_i} \quad (3)$$

In conclusion, by computing a checking model for each system behavioral-mode, Theorem 1 and Theorem 2 give an explicit method to compute if a faulty behavioral mode is not detectable and if a behavioral mode is not isolable from another behavioral mode. The algorithm presented later will be based on these results.

3.3 Isolability and Checking Models

There might exist several checking models of a system behavioral-mode b_j as seen previously. Assume that one checking model $C^1_{b_j}$ is a proper subset of another checking model $C^2_{b_j}$, i.e. $C^1_{b_j} \subset C^2_{b_j}$. If $C^2_{b_j} \subseteq M_{b_i}$ then $C^1_{b_j} \subseteq M_{b_i}$ but the opposite does not hold. This and Theorem 2 imply that if checking model $C^2_{b_j}$ implies that b_i is not isolable from b_j then $C^1_{b_j}$ does that too. Now assume that $C^1_{b_j} \subset M_{b_i} \subset C^2_{b_j}$. By using $C^1_{b_j}$ as checking model for b_j, it is concluded from Theorem 2 that b_i is not isolable from b_j. However if $C^2_{b_j}$ is used as checking model then no conclusion can be drawn. Hence the strongest conclusion is given by the smallest checking model. By finding smaller checking models than M_b more faults can be concluded to not be isolable from others.

4 Finding Checking Models

The minimal checking models of a system behavioral-mode are unknown and depends on the analytical expressions of the equations in the model. A brute-force approach to compute the minimal checking models would be to compute observation sets for subsets of equations and compare it to the observation set of the behavioral model. Even for models of the size and complexity like the water-tank example, automatic computation of observation sets by using computer algebra, like for example Mathematica, is computationally demanding. For a large industrial example this approach would be computationally intractable. Instead of requiring an exact determination of all minimal checking models of b, we propose to compute the smallest checking model of b, that can be obtained with the structural method to be presented in Section 4.3. This model will in the continuation be called the smallest checking model for b. The strategy to find the smallest checking model of b will be to start with the corresponding behavioral model and remove equations which are not needed to define the observation set for the behavioral model, i.e. to remove non-monitorable equations.

4.1 Excluding Non-monitorable Equations

If X is any set of variables, then $\mathbf{x}$ will denote the vector of the variables in X. If M is a set of equations with variables X then $M(\mathbf{x})$ will denote the conjunction of the analytical equations in M where the values of the variables X are set to $\mathbf{x}$. Consider a set of equations M with unknown variables X and known variables Z. If X is partitioned into X_1 and X_2 and

$$\forall \mathbf{z} \forall \mathbf{x_2} \exists \mathbf{x_1} : M(\mathbf{x_1}, \mathbf{x_2}, \mathbf{z}) \tag{4}$$

then the set M of equations is said to be X_1-*satisfiable*. For example, let $M = \{e_3\}$ and $X_1 = \{w\}$. For arbitrary values of f_t and q_2 there exists a value $w = (1 - f_t)q_2^2$ such that e_3 is true, i.e. $\{e_3\}$ is $\{w\}$-satisfiable.

Theorem 3 *If a model $M \subseteq M_b$ is X_1*-satisfiable *and no variable in X_1 is contained in $M_b \backslash M$, then $M_b \backslash M$ is a checking model of b.*

An alternative formulation of Theorem 3 is that if M is X_1-*satisfiable* and no variable in X_1 is contained in $M_b \backslash M$, then M is non-monitorable. This means that a checking model smaller than the behavioral model can be computed by removing equation set M from the behavioral model M_b. To give an example of how this is done, consider the behavioral mode $\mathbf{W}$ for the water-tank example. Since $\{e_3\}$ is $\{w\}$-satisfiable and e_3 is the only equation in $M_{\mathbf{W}}$ where w is included, $M_{\mathbf{W}} \backslash \{e_3\}$ is a checking model of $\mathbf{W}$ according to Theorem 3. In [Blanke *et al.*, 2003; Frisk *et al.*, 2003; Pulido and Alonso, 2002] analytical assumptions imply that the minimal checking model for a behavioral mode b is equal to the equations included in the vertical tail M_b^+ of the Dulmage-Mendelsohn decomposition of the behavioral model M_b. The smallest checking model that can be derived using Theorem 3 is not related to M_b^+.

4.2 Structural Method

A structural method will be used to compute non-monitorable equation sets for a behavioral model. The structure of a model is an abstraction of the model in the sense that it includes which variables that are included in each equation [Cassar and Staroswiecki, 1997]. The structure of the water-tank model in Table 1 is shown in Table 2 as a *biadjacency matrix* [Asratian *et al.*, 1998]. An "X" or an "O" in row e and column x means that x is included in e. An entry corresponding to equation e and variable x is marked "X" if $\{e\}$ is $\{x\}$-satisfiable and otherwise "O". Insights of the physics can be used to specify where to put "X":s.

By using this additional information together with the structure it is possible to find non-monitorable equation sets with cardinality one as follows. If e is the only equation in M_b that contains a variable x and this variable is marked with an "X" in the biadjacency matrix, then $\{e\}$ satisfies the conditions in Theorem 3, i.e. $\{e\}$ is non-monitorable. The next theorem will give theoretical results needed for computing non-monitorable equation set with cardinality greater than 1.

Theorem 4 *Let M_1 and M_2 be disjoint sets of equations. If M_1 is X_1-satisfiable, M_2 is X_2-satisfiable, and does not contain any variable in X_1, then it follows that $M_1 \cup M_2$ is $(X_1 \cup X_2)$-satisfiable.*

This theorem provides a recursive computation of a non-monitorable set of equations M that satisfies Theorem 3. To exemplify Theorem 3 consider the behavioral model $M_b = M_{\mathbf{PW}}$ in the water-tank example. The model $M_{\mathbf{PW}}$ consists of all equations in Table 1 except for e_1, e_7, and e_8. The model $M_1 = \{e_2\}$ is $\{q_1\}$-satisfiable and $M_2 = \{e_4\}$ is $\{w'\}$-satisfiable. Now, since $\{e_4\}$ and $\{e_2\}$ are disjoint and q_1 is not included in e_4, Theorem 4 implies that $\{e_2, e_4\}$ is $\{q_1, w'\}$-satisfiable. Furthermore, the variables in $\{q_1, w'\}$ are not included in $M_{\mathbf{PW}} \backslash \{e_2, e_4\}$ which means that $M_{\mathbf{PW}} \backslash \{e_2, e_4\}$ is a checking model of $\mathbf{PW}$, according to Theorem 3. In this way, it is possible to find the smallest checking model by finding a non-monitorable equation and remove them from the model.

Table 2: The structure of the model in Table 1.

Equation	Unknowns									Knowns				
	q_1	w	w'	q_2	q_2'	f_t	f_t'	f_q	f_q'	u	y_w	$\dot{y}_w$	y_q	$\dot{y}_q$
e_1	X									X				
e_2	X		X	X										
e_3		X		O		O								
e_4			X	O	O	O	O							
e_5							X							
e_6						X								
e_7		X									X			
e_8			X									X		
e_9				X				X					X	
e_{10}					X				X					X
e_{11}									X					
e_{12}								X						

4.3 Algorithm

Next we will present a recursive algorithm for computing the smallest possible checking model of a behavioral mode b given the type of information given in Table 2. The input to the algorithm is a structure as the one shown in Table 2 with "O":s and "X":s.

Algorithm 1 `FindCheckingModel`
input: *The structure of M_b.*

```
    if there exists an $e \in M_b$ with an unknown $x$ only in $e$
    and the entry $(e, x)$ is marked "$X$" do
        $M_b$ = FindCheckingModel($M_b \backslash \{e\}$);
    end if
return: The checking model $M_b$.
```

The correctness of the algorithm is implied by Theorem 3 and Theorem 4. For a checking model C_b obtained by Algorithm 1, it holds that $M_b^+ \subseteq C_b \subseteq M_b$. Note that the output model of Algorithm 1 contains all algebraic loops contained in the input model. However, by deriving a checking model using Theorem 3 and Theorem 4 directly, not all equations containing algebraic loops need to be contained in the checking model.

Consider for the water-tank example the behavioral mode $\mathbf{PW}$. The structure is seen in Table 2. `FindCheckingModel` is first called with input $M_b = M_{\mathbf{PW}}$. The variable q_1 is among the equations in $M_{\mathbf{PW}}$ only included in e_2 and the corresponding entry is marked "X", i.e. the if-condition is satisfied and `FindCheckingModel` is called with input $M_{\mathbf{PW}} \backslash \{e_2\}$. Now the if-condition is also satisfied, because w is only included in e_3 and (e_3, w) is marked "X". Continuing the recursion in this way `FindCheckingModel`$(M_{\mathbf{PW}})$ returns the empty set $\varnothing$ which is the checking model of $\mathbf{PW}$ to be used in the isolability computation later. This means that $\mathbf{PW}$ is always a diagnosis.

5 Isolability Prediction Algorithm

Algorithm 1 computes the smallest checking model C_{b_j} of a behavioral mode b_j given the structure of the behavioral model M_{b_j}. If (3) is true for the computed checking model C_{b_j} and a behavioral model M_{b_i} of another behavioral mode b_i, Theorem 2 implies that b_i is not isolable from b_j. This is the idea used in the next algorithm for computing behavioral modes that are not isolable from other behavioral modes. Let $\mathcal{B}$ be the set of all system behavioral-modes and let $\mathcal{I} \subseteq \mathcal{B} \times \mathcal{B}$ be a set of pairs of behavioral modes (b_i, b_j) such that if $(b_i, b_j) \in \mathcal{I}$ then b_i is not isolable from b_j.

Algorithm 2 `IsolabilityPrediction`
input: *The structure of a diagnostic model and a set of system behavioral-modes $\mathcal{B}$.*

```
$\mathcal{I} = \varnothing$;
for all $b_j \in \mathcal{B}$ do
    $C_{b_j}$ = FindCheckingModel($M_{b_j}$);
    for all $b_i \in \mathcal{B}$ do
        if $C_{b_j} \subseteq M_{b_i}$ do
            $\mathcal{I} = \mathcal{I} \cup \{(b_i, b_j)\}$;
        end if
    end for
end for
return: $\mathcal{I}$
```

Algorithm 2 computes the largest set $\mathcal{I}$ that can be derived using only the type of information given in Table 2. The interpretation of the output of the algorithm is discussed in the next section. The purpose of Algorithm 2, as stated here, is to illustrate the idea and not to explain additional features that can lower the computational complexity. However one such improvement is to use the fact that $M_{b_i} \subseteq M_{b_j}$ implies $C_{b_i} \subseteq C_{b_j}$ and in each step compute a checking model for a maximal behavioral model.

5.1 Isolability Prediction Interpretation

The isolability property can be seen as a partial order on the set of equivalence classes generated by mutually not isolable behavioral modes. Two equivalence classes of behavioral modes B_i and B_j are related as $\leq$ when for all $b_i \in B_i$ and for all $b_j \in B_j$, b_i is not isolable from b_j. Figure 3 shows the partial order computed by Algorithm 2 when all multiple faults of the water-tank process are considered. For example the four behavioral modes in the top are an equivalence class and are therefore not isolable from each other. In Figure 3 it can also be seen that no fault is isolable from faults with a superset of faulty components. This is not surprising since no equation in the model holds only in a faulty behavioral mode. Furthermore, since the top element is an upper bound for all behavioral modes, it means that these faults will always be diagnoses, in fact they all have the empty set as their checking models.

6 Illustrative Examples

Previous sections have described Algorithm 2 that predicts the isolability. Here, two examples illustrate how Algorithm 2 can be used.

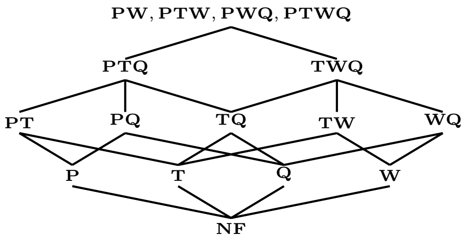

Figure 3: An isolability prediction of the water-tank process.

6.1 Fault Modeling Specification

Assume safety or legislative requirements state demands on the fault isolability. Given a diagnostic model including fault models, it can be determined by applying Algorithm 2 to the diagnostic model if the proposed fault modeling is insufficient for the fault isolability demands.

Assume that all double faults must be isolable from each other in the water-tank process. The result shown in Figure 3 implies that the isolability demands cannot be fulfilled with the proposed model in Table 1. For example no double fault is isolable from $\mathbf{PW}$. To make any behavioral mode isolable from $\mathbf{PW}$ the behavioral model $M_{\mathbf{PW}}$ must be improved for example by additional fault modeling. The faulty components in $\mathbf{PW}$ are the pump P and the water-level sensor W and non of these components have fault models.

Assume that it is reasonable to use a constant bias fault model for the water-level sensor. Let f_w be the size of the bias fault. Equation e_7 can now be replaced by $y_w = w + f_w$ and e_8 by $\dot{y}_w = w' + f'_w$ which both hold in any system behavioral-mode. Furthermore, the new equations $e_{13} : f_w = 0$ which holds when $W = \text{NF}$ and $e_{14} : f'_w = 0$ which always is true are added to the model in Table 1. By applying Algorithm 2 to the model including the new fault model, a smaller set $\mathcal{I}$ is obtained. This means that some faults that were not isolable from some other faults without the fault model, now might be isolable. The result with the additional fault model is that it might be possible to isolate all double faults from all other double faults. For this example it is also possible to analyze the true isolability using the analytical expressions. For example consider the behavioral modes $\mathbf{PW}$ and $\mathbf{PT}$. Without the additional fault model, $\mathbf{PT}$ was not isolable from $\mathbf{PW}$. When including the fault model the observation set $\mathcal{O}(M_{\mathbf{PW}})$ for $\mathbf{PW}$ is defined by $\dot{y}_w - 2\, y_q\, \dot{y}_q = 0$ and $\mathcal{O}(M_{\mathbf{PT}})$ is defined by $\dot{y}_w\, y_q - 2\, y_w\, \dot{y}_q = 0$ and if $y_q = 0$ then $y_w = \dot{y}_w = 0$. Both these expressions can be computed by elimination of all unknowns in their corresponding checking models respectively. Since these checking models are smaller than the corresponding behavioral model, the elimination problem is reduced. The mode $\mathbf{PT}$ is isolable from $\mathbf{PW}$ if $\mathcal{O}(M_{\mathbf{PT}}) \setminus \mathcal{O}(M_{\mathbf{PW}}) \neq \varnothing$. An example of observations in $\mathcal{O}(M_{\mathbf{PT}}) \setminus \mathcal{O}(M_{\mathbf{PW}})$ is $y_q \neq 0$, $\dot{y}_q \neq 0$, $y_w \neq y_q^2$, and $\dot{y}_w = 2 y_w\, \dot{y}_q / y_q$. Hence $\mathcal{O}(M_{\mathbf{PT}}) \setminus \mathcal{O}(M_{\mathbf{PW}}) \neq \varnothing$, i.e. $\mathbf{PT}$ is isolable from $\mathbf{PW}$. According to the result of Algorithm 2, it is possible that all double faults are isolable from all other double faults and it can be shown to be so.

6.2 Design Alternative Selection

Suppose there are different design alternatives, e.g. different possible sensor configurations. Since only a course model is needed as input to Algorithm 2, the isolability aspects of different design alternatives can easily be evaluated.

Let the isolability demands be the same as in the previous section and assume that there are two design alternatives for the water-tank process, one as described in Section 2 and one with an additional flow sensor Q_{extra} measuring q_1. We know from the previous discussion that it is not possible to isolate all double faults from each other by using the model in Figure 1. The result of applying Algorithm 2 to an extended model including the additional sensor Q_{extra} answers the question if the model with the additional sensor can be sufficient to meet the isolability demands.

The extended model is obtained by adding the equation $e_{13} : y = q_1$ with the assumption $Q_{\text{extra}} = \text{NF}$. Note that an extra sensor will change the set of all system behavioral modes. In this example the number of components is 5 and the original model has only 4 components. By including the additional sensor, all double faults, including the new once introduced by Q_{extra}, might be isolable from any other double fault according to the result of Algorithm 2. Analytical analysis can be done as in Section 6.1 to conclude that all double faults are isolable from all other double faults.

To summarize the results of the examples, without any fault model or any additional sensor, this analysis shows that there are double faults which are not isolable from other double faults. However, by adding the proposed fault model or the water-level sensor it can be shown that all double faults are isolable from all other double faults.

7 Conclusions

In the development of processes including diagnosis, design decisions are taken, e.g. sensor configuration selection, which affects the fault isolability possibilities. This paper has presented an algorithm and a methodology that easily can be applied to different design alternatives and evaluate their isolability limitations.

The framework from [Dressler *et al.*, 1993; Nyberg and Krysander, 2003], which handles general fault models, has been used. In [Travé-Massuyès *et al.*, 2003; Frisk *et al.*, 2003], and [Blanke *et al.*, 2003] assumptions are made such that all detectable faults violate the over-constrained subsystem. Here a more careful assumption is made. The advantage of being careful is that in contrast to the results in [Travé-Massuyès *et al.*, 2003], [Frisk *et al.*, 2003], and [Blanke *et al.*, 2003] no analytical assumptions need to be satisfied to draw the conclusions about the detectability or the isolability.

Algorithm 2 computes faults that are not isolable from others by using the structure of a diagnostic model as the one in Table 2. This was done by combining Algorithm 1, which computes the smallest checking models that can be computed by using structural models as the one in Table 2, and the link between checking models and isolability stated in Theorem 2. Furthermore, in Section 6.1 it was shown how Algorithm 2 could detect insufficient fault modeling. The analysis revealed faults not isolable from other faults and by the example a methodology was proposed to locate required additional fault modeling. Section 6.2 showed how Algorithm 2 could be used to find the isolability limitations of different design alternative for a process to be diagnosed.

In conclusion, it is believed that structural methods for isolability analysis have an advantage of analytical methods to support decisions early in the design process. The proposed algorithm is the only structural algorithm which computes faults that are not isolable from others without any analytical assumptions.

References

[Asratian *et al.*, 1998] Armen Asratian, Tristan Denley, and Roland Häggkvist. *Bipartite Graphs and their Applications*. Cambridge University Press, 1998.

[Blanke *et al.*, 2003] M. Blanke, M. Kinnert, J. Lunze, and M. Staroswiecki. *Diagnosis and Fault-Tolerant Control*. Springer-Verlag, 2003.

[Cassar and Staroswiecki, 1997] J. P. Cassar and M. Staroswiecki. A structural approach for the design of failure detection and identification systems. In *IFAC Control of Industrial Systems*, Belford, France, 1997.

[Dressler *et al.*, 1993] O. Dressler, C. Böttcher, M. Montag, and A. Brinkop. Qualitative and quantitative models in a model-based diagnosis system for ballast tank systems. In *Int. Conf. on Fault Diagnosis (TOOLDIAG)*, pages 397–405, Toulouse, France, 1993.

[Frisk *et al.*, 2003] Erik Frisk, Dilek Dştegör, Mattias Krysander, and Vincent Cocquempot. Improving fault isolability properties by structural analysis of faulty behavior models: application to the DAMADICS benchmark problem. In *Proceedings of IFAC Safeprocess0́3*, Washington, USA, 2003.

[Krysander and Nyberg, 2002] Mattias Krysander and Mattias Nyberg. Structural analysis utilizing MSS sets with application to a paper plant. In *Proc. of the Thirteenth International Workshop on Principles of Diagnosis*, Semmering, Austria, May 2002.

[Krysander and Nyberg, 2005] Mattias Krysander and Mattias Nyberg. Fault isolability prediction of diagnositic models. Technical Report LiTH-ISY-R-2678, Dept. of Electrical Engineering Linköpings universitet, 2005. URL:http://www.vehicular.isy.liu.se/Publications/.

[Mattson and Söderlind, 1993] S. Mattson and G. Söderlind. Index reduction in differential-algebraic equations using dummy derivatives. *SIAM Journal on Scientific Computing*, 14(3):677–692, 1993.

[Nyberg and Krysander, 2003] Mattias Nyberg and Mattias Krysander. Combining AI, FDI, and statistical hypothesis-testing in a framework for diagnosis. In *Proceedings of IFAC Safeprocess0́3*, Washington, USA, 2003.

[O. Dressler, 2003] P. Struss O. Dressler. A toolbox integrating model-based diagnosability analysis and automated generation of diagnostics. In *Proceedings of the 14th International Workshop on Principles of Diagnosis (DX03)*, pages 99–104, Washington, USA, 2003.

[Pulido and Alonso, 2002] B. Pulido and C. Alonso. Possible conflicts, ARRs, and conflicts. In M. Stumptner and F. Wotawa, editors, *Proceedings DX-2002*, pages 122–127, Semmering, Austria, 2002.

[Travé-Massuyès *et al.*, 2003] L. Travé-Massuyès, T. Escobet, and S. Spanache. Diagnosability analysis based on component supported analytical redundancy relations. In *Proceedings of IFAC Safeprocess'03*, Washington, USA, 2003.