Hybrid automaton incremental construction for online diagnosis

Jorge Vento\textsuperscript{1}, Louise Travé-Massuyès\textsuperscript{2,3}, Ramon Sarrate\textsuperscript{1} and Vicenç Puig\textsuperscript{1}

\textsuperscript{1}Advanced Control Systems (SAC), Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
\textsuperscript{2}CNRS, LAAS, 7, avenue du Colonel Roche, F-31400 Toulouse, France
\textsuperscript{3}University of Toulouse, LAAS, F-31400 Toulouse, France
\textsuperscript{1}e-mail: \{jorge.isaac.vento,ramon.sarrate,vicenc.puig\}@upc.edu
\textsuperscript{2}e-mail: louise@laas.fr

Abstract

This paper proposes a method to track the system mode and diagnose a hybrid system without building an entire diagnoser off-line. The method is supported by a hybrid automaton model that represents the hybrid system continuous and discrete behavioral dynamics. Diagnosis is performed by interpreting the events and measurements issued by the physical system directly on the hybrid automaton model. This interpretation leads to building the useful parts of the diagnoser incrementally, developing only the branches that are required to explain the occurrence of the incoming events. The resulting diagnoser adapts to the system operational life and is much less demanding in terms of memory storage. The proposed framework subsumes previous works in that it copes with both structural and non-structural faults. The method is validated on an application case study based on the sewer network of the Barcelona city.

1 Introduction

The approaches to detect and isolate faults in hybrid systems have been addressed by both the FDI and DX communities. In the FDI approach, the diagnosis is based on the hybrid automaton to track the system mode [Bayoudh \textit{et al.}, 2008; Vento \textit{et al.}, 2011] combining the continuous and discrete techniques to detect and isolate faults. On the other hand, in the DX approach some authors have proposed alternative ways to diagnose hybrid systems like using the hybrid bond graph formalism [Narasimhan and Biswas, 2007; Daigle, 2008].

This paper follows the work presented in [Vento \textit{et al.}, 2011; Bayoudh \textit{et al.}, 2008] where parity-space residuals are used to track the mode and diagnose the hybrid system. In [Vento \textit{et al.}, 2011], the operation modes represent nominal behavior and diagnosis focuses on fault detection and isolation of non-structural faults, i.e. faults that do not change the structure of the model (e.g. additive faults in sensors and actuators). In [Bayoudh \textit{et al.}, 2008], the operation modes may be nominal or faulty, leading to the capability of detecting and isolating structural faults (e.g. an actuator stuck at a given position, opened or closed). In both cases, a set of analytical redundancy relations (ARR) are inferred from the set of equations in each mode and they are used to generate a set of residuals. In the case of non-structural faults, the fault effect on the residuals of every mode is assumed to be known and is captured by theoretical fault signatures. Tracking the system mode involves detecting that the residuals of the current mode are different from zero and checking the theoretical fault signatures against the residuals evaluated with measurements. In the case of structural faults, fault models are assumed to be known and the residuals of a faulty mode are expected to become zero when the fault is present.

The methods presented in the above mentioned works rely on a finite state machine called a diagnoser [Sampath \textit{et al.}, 1995] which is built off-line from the hybrid model and the residuals are generated for each mode as explained in [Vento \textit{et al.}, 2011; Bayoudh \textit{et al.}, 2008]. The main issue with these off-line approaches is that since the number of states of the diagnoser grows exponentially with the number of states of the hybrid automaton, the generation of the set of residuals for every mode may be a limiting factor.

This paper proposes a method to track the system mode and diagnose the hybrid system without building the entire diagnoser off-line. Diagnosis is performed by interpreting the events and measurements issued by the physical system directly on the hybrid automaton model. This interpretation leads to building the useful parts of the diagnoser incrementally, developing only the branches that are required to explain the occurrence of the incoming events. Generally, a hybrid system operates in a small region compared to the entire behavioral space defined by the hybrid automaton states. A significant gain hence comes from the proposed approach. Moreover, the proposed framework subsumes previous works in the sense that structural and non-structural faults are considered at the same time.

The structure of the paper is the following. In Section 2, the hybrid model is presented. Section 3 provides the principles of the proposed method to diagnose faults in hybrid systems. In Section 4, the method to incrementally build the diagnoser of the hybrid system is presented as well as its implementation. In Section 5, an application case study based on the sewer network of the Barcelona city is used to assess the validity of the proposed approach. Finally, conclusions are given in Section 6.
2 Hybrid System Modeling

The hybrid automaton model results from an adaptation of [Lygeros et al., 2003; Bayoudh et al., 2008; Vento et al., 2011]. This work assumes linear continuous dynamics in each mode represented by discrete-time state space models. Let us consider that the model of the hybrid system to be diagnosed can be described by the following hybrid automaton $HA = \langle Q, \mathcal{X}, \mathcal{U}, \mathcal{Y}, \mathcal{F}, \mathcal{G}, \mathcal{H}, \Sigma, \mathcal{T} \rangle$, where:

- $Q$ is a set of modes. Each $q_i \in Q$ with $|Q| = n_q$ represents a nominal operation or faulty mode of the system such that $Q = Q_N \cup Q_F$.
- $q_0 \in Q$ is the initial mode.
- $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ defines the continuous state space. $\mathbf{x}(k) \in \mathcal{X}$ is the discrete-time state vector and $\mathbf{x}_0$ is the initial state vector.
- $\mathcal{U} \subseteq \mathbb{R}^{n_u}$ defines the discrete-time input space. $\mathbf{u}(k) \in \mathcal{U}$ is the discrete-time input vector.
- $\mathcal{Y} \subseteq \mathbb{R}^{n_y}$ defines the continuous output space. $\mathbf{y}(k) \in \mathcal{Y}$ is the discrete-time output vector.
- $\mathcal{F}$ is the set of faults that can be partitioned as $\mathcal{F} = \mathcal{F}_s \cup \mathcal{F}_u$ taking into account structural and non-structural faults.
- $G$ defines a set of discrete-time state affine functions for each mode:
  $$\mathbf{x}(k+1) = A_{\mathbf{x}} \mathbf{x}(k) + B_{\mathbf{x}} \mathbf{u}(k) + F_{\mathbf{x}} \mathbf{f}_{\mathbf{u}}(k) + E_{\mathbf{x}} f_{\mathbf{u}}(k) \quad (1)$$
  where $A_{\mathbf{x}} \in \mathbb{R}^{n_x \times n_x}, B_{\mathbf{x}} \in \mathbb{R}^{n_x \times n_u}$ and $E_{\mathbf{x}} \in \mathbb{R}^{n_x \times 1}$ are the state matrices in mode $q_i$. $\mathbf{f}_{\mathbf{u}}(k)$ is a vector representing non-structural faults with $F_{\mathbf{x}}$ being the fault distribution matrix. The case $f_{\mathbf{u}}(k) = 0$ corresponds to a nominal behaviour.
- $\mathcal{H}$ defines a set of discrete-time output affine functions for each mode:
  $$\mathbf{y}(k) = C_{\mathbf{y}} \mathbf{x}(k) + D_{\mathbf{y}} \mathbf{u}(k) + F_{\mathbf{y}} \mathbf{f}_{\mathbf{u}}(k) + E_{\mathbf{y}} f_{\mathbf{u}}(k) \quad (2)$$
  where $C_{\mathbf{y}} \in \mathbb{R}^{n_y \times n_x}, D_{\mathbf{y}} \in \mathbb{R}^{n_y \times n_u}$ and $E_{\mathbf{y}} \in \mathbb{R}^{n_y \times 1}$ are the output matrices in mode $q_i$ and $F_{\mathbf{y}}$ is the fault distribution matrix.
- $\Sigma = \Sigma_\sigma \cup \Sigma_{\mathbf{c}} \cup \Sigma_{\mathcal{F}_s}$ is the set of events. Spontaneous mode switching events ($\Sigma_\sigma$), input events ($\Sigma_{\mathbf{c}}$) and structural fault events ($\Sigma_{\mathcal{F}_s}$) are considered. $\Sigma$ can be partitioned as $\Sigma_\sigma \cup \Sigma_{\mathbf{u}}$ where $\Sigma_\sigma$ represents the set of observable events and $\Sigma_{\mathbf{u}}$ represents the set of unobservable events. $\Sigma_{\mathcal{F}_s} \subseteq \Sigma_{\mathbf{u}}, \Sigma_{\mathbf{c}} \subseteq \Sigma_\sigma$ and $\Sigma_\sigma$ may have elements in both sets.
- $\mathcal{T} : Q \times \Sigma \rightarrow Q$ is the transition function. The transition from mode $q_i$ to mode $q_j$ labeled with an event $\sigma \in \Sigma$ is denoted by $\mathcal{T}(q_i, \sigma) = q_j$ or by $t_{ij}$ when the event is of no interest.

Alternatively, the model given by (1)-(2) can be expressed in input-output form using the delay operator which is denoted by $p^{-1}$ and considering zero initial conditions as follows:

$$\mathbf{y}(k) = \mathbf{M}_i(p^{-1}) \mathbf{u}(k) + \mathbf{T}_i(q^{-1}) \mathbf{f}_{\mathbf{u}}(k) + \mathbf{E}_{\mathbf{m}_i}(p^{-1}) \quad (3)$$

where $p^{-1}$ is the delay operator, $\mathbf{M}_i(p^{-1})$ represents the transfer function between inputs and outputs of the system, $\mathbf{T}_i(p^{-1})$ is the non-structural fault transfer function and $\mathbf{E}_{\mathbf{m}_i}(p^{-1})$ is a constant term.

3 Proposed Hybrid Diagnosis Method

3.1 Principles of the Method

Model-based diagnosis is based on the use of a model of the monitored dynamic system to detect and isolate faults. The estimated system behaviour obtained from the system model is compared with the real behaviour available through sensor measurements. In particular, FDI algorithms for hybrid systems take into account which is the current operation mode to generate the set of residuals, used to build consistency indicators, and to achieve the diagnosis task [Vento et al., 2011]. The scheme of the proposed method to diagnose hybrid systems is shown in Fig. 1. The figure shows the different tasks involved in online diagnosis. The original idea is to build an hybrid diagnoser in an incremental manner when an event occurs.

Figure 1: Conceptual block diagram for the proposed methodology

The method consists in tracking the mode sequence synchronously thanks to a diagnoser which is built incrementally from the so-called behaviour automaton, considering the possible current modes of the system and their successors. The original idea of the paper is to build the diagnoser in an incremental manner when an event occurs. The behaviour automaton includes so called signature events that abstract the residual behaviors. The transitions labelled by unobservable events in $HA$ may hence turn observable by means of the signature events thanks to the discernability property (cf. section III.B).

To detect and isolate faults in the system two possibilities exist. On one hand, structural faults which produce changes in the dynamics are included in $HA$ as faulty modes with their own dynamic model. Therefore, the corresponding mode is recognized when its consistency indicators are in agreement with measurements. On the other hand, non-structural faults are represented as disturbances on the models of the different modes of $HA$. Using the fault sensitivity of these models, a fault signature matrix can be generated. Then, a consistency test using this matrix is carried out comparing the set of observed consistency indicators with the columns of the fault signature matrix.
Diagnosis is based on the single fault assumption during the detection phase. However, two faults can occur sequentially, as long as the first one corresponds to a structural fault and the second one to a non-structural fault. Moreover, it is assumed that there is a minimal time between state transitions according to the dwell time of $HA$.

### 3.2 Diagnosis Based on Residual Consistency Indicators

Diagnosis based on continuous dynamics relies on residual properties. The set of residuals for the mode $q_i$ is given by:

$$r_i(k) = y(k) - G_i(p^{-1})u(k) - H_i(p^{-1})y(k) - E_i(p^{-1})$$

(4)

where $G_i(p^{-1})$, $H_i(p^{-1})$ and $E_i(p^{-1})$ can be calculated for instance, using the parity space or observer approach (for more details see e.g. [Ding et al., 2008])

Once the residuals have been generated, they are evaluated with the measurements against a threshold, providing one consistency indicator of the following form for each residual:

$$\varphi_i^l(k) = \begin{cases} 0 & \text{if } |r_i^l(k)| \leq \tau_i^l \\ 1 & \text{if } |r_i^l(k)| > \tau_i^l \end{cases}$$

(5)

where $l \in \{1, \ldots, n_r\}$, $n_r$ is the number of residuals for mode $q_i$ and $\tau_i^l$ is the threshold associated with the residual $r_i^l(k)$. The consistency indicators are then gathered in the vector $\Phi_q(k) = [\varphi_i^l(k), \ldots, \varphi_i^{n_r}(k)]$. Sumarizing, consistency indicator vector $\Phi_q(k)$ is built from the binarised residuals (5) of mode $q_i$, evaluated with the measurements corresponding to the current mode of the system at time $k$.

An important property to track the system mode is discernability. Discernability between two modes is the property that assesses whether two modes can be distinguished based on continuous measurements. If two modes $q_i$ and $q_j$ are discernable and the system changes from mode $q_i$ to mode $q_j$ or viceversa, the sequence of signals $(u(k), y(k))$ change from being consistent with mode $q_i$ to being consistent with mode $q_j$ or viceversa. This property can be verified using the consistency indicators defined above [Mezyani, 2007]: two modes are discernable iff the set of consistency indicators satisfy $\Phi_{q_i}(k) \neq \Phi_{q_j}(k)$ with measurements corresponding to mode $q_i$ and viceversa.

If neither the consistency indicators of mode $q_i$ nor those of mode $q_j$ are in agreement with measurements it is assumed that a non-structural fault is affecting the system. This kind of faults are identified using the concept of fault sensitivity [Vento et al., 2011], which is determined by the expression:

$$\Lambda_i(p^{-1}) = (I - H_i(p^{-1}))Y_i(p^{-1})$$

where $Y_i(p^{-1})$ represents the non-structural fault transfer function between the input and the non-structural fault vector in (1)-(2).

In particular, given the fault sensitivity of the $j^{th}$ residual with respect to the $l^{th}$ fault denoted as $\Lambda_i(j,l)$ (i.e. the element $(j,l)$ of the sensitivity matrix $\Lambda_i$), the element $(j,l)$ of the fault signature matrix is determined as follows:

$$FS_i(j,l) = \begin{cases} 1 & \text{if } \Lambda_i(j,l)(p^{-1}) \neq 0 \\ 0 & \text{if } \Lambda_i(j,l)(p^{-1}) = 0 \end{cases}$$

(6)

i.e., if the $j^{th}$ residual in mode $q_i$ depends on the $l^{th}$ fault, it is coded as a 1 and it is coded as a 0 otherwise. A non-structural faulty situation is detectable, i.e. discernible from the nominal mode, is its signature is different from 0. Two faulty situations corresponding to the $l^{th}$ and the $l'$th non-structural faults are discernible iff their signatures are different, i.e. $FS_i(j,l) \neq FS_i(j,l')$. Since in a hybrid system, residuals change with the mode, the fault sensitivity as well as the theoretical fault signature matrix depend on the mode.

If the situation is such that neither a structural fault nor a non-structural fault can be isolated, the system mode is assumed unknown. The diagnosis based on consistency indicators assumes that the residual dynamics have time to establish between two consecutive transitions.

### 4 Hybrid Diagnosis

The diagnoser of the hybrid system is a finite state machine built from the behavior automaton and used, on one hand, to perform on-line diagnosis and on the other hand, to check the diagnosability of the hybrid system as presented in [Bayoudh et al., 2008]. The method proposes to incrementally build the hybrid diagnoser from the behaviour automaton obtained while the system is monitored.

#### 4.1 Behaviour Automaton

The behaviour automaton is the finite state generator of the abstract language $L(HA)$ resulting from abstracting the continuous dynamics captured by the residual consistency indicators in terms of discrete signature-events [Bayoudh et al., 2008]. The behaviour automaton is defined by $B = \langle Q, \Sigma, \tau, q_0 \rangle$:

- $Q = Q_0 \cup Q_{\text{fs}}, \cup Q_{\text{ns}}, \cup Q_{\text{fr}}, \cup Q_{\text{frn}}$ is the set of discrete states where:
  - $Q_0$ is the set of system modes,
  - $Q_{\text{fs}}$ is the set of transient modes between two discernible modes in $HA$,
  - $Q_{\text{fr}}$ is the set of transient modes to represent a non-structural fault occurrence,
  - $Q_{\text{frn}}$ is the set of modes representing non-structural fault behaviours.

- $\tau_0$ is the initial state,

- $\Sigma = \Sigma_0 \cup \Sigma_{\text{fr}} \cup \Sigma_{\text{frn}} \cup \Sigma_{\text{frn}}$ is the set of events where:
  - $\Sigma_0$ is the set of system events,
  - $\Sigma_{\text{fr}}$ is the set of signature-events generated by function $f_{S\text{fr}},\text{ev}$ defined by (7),
  - $\Sigma_{\text{frn}}$ is the set of fault events related to a non-structural fault occurrence,
  - $\Sigma_{\text{frn}}$ is the set of signature-events for non-structural faults generated by function $f_{S\text{frn},\text{ev}}$ defined by (7),

- $\tau: Q \times \Sigma \rightarrow Q$ is the partial transition function of the behaviour automaton.

In this paper, it is proposed to build $B$ incrementally following Algorithm 1, which is an adaptation of the previous approach proposed in [Vento et al., 2011]. The algorithm explores $HA$ taking into account only the modes in which the real system is possibly operating at time instant $k$.

Assuming that the system is possibly operating in a given mode or set of modes denoted by $q_D$, to build incrementally
Algorithm 1. Builder($q_D$)

1: Create a queue $Q$
2: for all $q_i \in q_{D0}$ do
3: Enqueue $q_i$ onto $Q$
4: end for
5: while $Q$ is not empty do
6: dequeue $q_i$ from $Q$
7: for all $q_j \in Succss(q_i)$ do
8: if $q_j \notin \Sigma\sigma$ then
9: $\Sigma = \{q_j\} \cup \Sigma$
10: Compute residual expression $r_j$ (∗)
11: Classify $q_j$ into $Q_{disc}$, $Q_{ns}$, then
12: Calculate FS$_{\delta q_j}$ associated to $F_{\nu_j}$
13: Determine the subsets of detectable faults $F_{\nu_j}^{+}$
14: Determine the set of non-detectable faults $F_{\nu_j}^{-}$
15: end if
16: end for
17: Define $\Sigma$ such that $T(q_i, \sigma) = q_j$:
18: if $\delta_{q_i, q_j} \notin \Sigma$ then
19: $\Sigma = \{\sigma\} \cup \Sigma$
20: else
21: if $q_i$ and $q_j$ are discernible then
22: $\Sigma^{+} = \{q_{j}'\} \cup \Sigma'$
23: else
24: if $q_i$ and $q_j$ are discernible then
25: $\Sigma^{+} = \{q_{j}'\} \cup \Sigma'$
26: else
27: $\Sigma^{+} = \{q_{j}'\} \cup \Sigma'$
28: end if
29: end if
30: if $\delta_{q_i, q_j} \notin \Sigma^{+}$ then
31: $\Sigma^{+} = \{q_{j}'\} \cup \Sigma'$
32: else
33: Enqueue $q_j$ onto $Q$
34: $T(q_i, \sigma) = q_j$
35: end if
36: end if
37: end for
38: for all $f_i \in F_{\nu_k}$ do
39: $\Sigma = \{q_{j}'\} \cup \Sigma'$
40: if $q_j \notin \Sigma^{+}$, then
41: $\Sigma^{+} = \{q_{j}'\} \cup \Sigma'$
42: end if
43: if $f_i \notin F_{\nu_j}$ then
44: $\Sigma = \{q_{j}'\} \cup \Sigma'$
45: end if
46: if $\delta_{q_i, q_j} \notin \Sigma^{+}$ then
47: $\Sigma^{+} = \{q_{j}'\} \cup \Sigma'$
48: end if
49: $T(q_i, \sigma) = q_j$
50: $\Sigma = \{q_{j}'\} \cup \Sigma'$
51: else
52: $T(q_i, \sigma) = q_j$
53: end if
54: end for
55: end while

B, all the successor modes for each mode of $q_D$ (Succss($q_i$)) are explored.

The set of explored modes is partitioned into subsets of non-discriminate-modes, forming the partition denoted by $Q_{disc} = Q_{dis} \cup \cdots \cup Q_{\nu_j}$. This information is stored in a knowledge base used by Algorithm 1 such that a new set of residuals is generated only when a mode has not been previously visited.

Transitions of $HA$ are integrated in $B$, evaluating the discernability property if necessary and analyzing the fault signature matrix to include non-structural faults as faulty modes. If a transition of $HA$ associated with an observable event is found in the exploration, it is kept in $B$ (see line 23). Otherwise, the discernability property is evaluated between these pair of modes. If the two modes are discernible then an intermediate mode is added between these modes. The outgoing transition is associated a signature-event, indicating that this mode change can be observed by means of the consistency indicators (see lines from 25-31).

To include information about non-structural faults in $B$, lines 38 to 49 of Algorithm 1 show how a fault signature matrix is generated associated with the current (set of) modes $q_D$. It is then analyzed to include the non-structural faulty modes $Q_{\nu_k}$ and the transient modes $Q_{\nu_j}$ based on discernability. Analyzing this matrix, the set of non-structural faults can be partitioned into detectable ($F_{\nu_j}^{+}$) and nondetectable ($F_{\nu_j}^{-}$) fault subsets. The set of detectable faults is further partitioned into: $F_{\nu_j}^{+} = \mathcal{F}_{1}^{1} \cup \cdots \cup \mathcal{F}_{N}^{1}$ (lines 13 to 15), where $N$ is the number of discernible, hence isolable, fault subsets.

The signature-events can represent both a mode change or a non-structural fault occurrence, according to this the signature-event can be labeled according to:

$$f_{Sig.ev} : Q \times \Sigma \mapsto \Sigma^{Sig}$$

$$f_{Sig.ev}(q_i, q_j) \mapsto \left\{ \begin{array}{ll}
\delta_{q_i, q_j} \in \Sigma^{Sig} & \text{if } (q_i, q_j) \text{ are discernible} \\
\delta_{q_i, q_j} \in \Sigma_{\nu_k}^{Sig} & \text{if } \text{FS}_{\nu_k}(\bullet, f_i) \neq 0
\end{array} \right.$$

where $q_i \in Q_{\nu_i}$ and $q_j \in Q_{\nu_j}$ with $Q_{\nu_i}, Q_{\nu_j} \subseteq Q_{disc}$, hence $\delta_{q_i, q_j}$ denotes an event associated with a mode change between modes in $HA$ and $\delta_{q_i, q_j}$ denotes an event associated with a fault $f_i$ belonging to the set $\mathcal{F}_{\nu_i} \subseteq \mathcal{F}_{\nu_k}$, and $q_j \in Q_{\nu_{\nu_k}}$ associated with the non-structural fault.

4.2 Hybrid Diagnoser

The diagnoser is a finite state machine $D = < Q_{D}, \Sigma_{D}, T_{D}, q_{D0} >$, where:

- $q_{D0} = \{q_0, \emptyset\}$ is the initial state of the diagnoser, which is assumed to correspond to a nominal system mode.
- $Q_{D}$ is the set of diagnoser states. An element $q_D \in Q_D$ is a set of the form $q_D = \{(q_i, l_i)\}$, where $q_i \in \Sigma$ and $l_i \in \Delta$, defines the set of system labels $\Delta_{SF} = \Delta_{F} \cup \Delta_{\nu_k}$, with $\Delta_{F} = \{f_1, \ldots, f_L\}$, and $\Delta_{\nu_k} = \{l_1, \ldots, l_M\}$ respectively. $\gamma, \mu \in \mathbb{Z}^+$. In $\Delta_{SF}$, $\emptyset$ represents the nominal behaviour.
- $\Sigma_{D} = \Sigma_{\nu_{\nu_k}}$ is the set of all observable events.
- $T_{D} : Q_{D} \times \Sigma_{D} \mapsto Q_{D}$ is the partial transition function of the diagnoser.

The transition function $T_{D}$ can be calculated according to the propagation algorithms explained in [Sampath et al., 1995], from the incremental $B$ obtained during system mode tracking. The algorithm to build the transition function is executed after the occurrence of an observable event whenever the state has not been previously visited. The part of the diagnoser obtained takes into account only the possible successor modes and hence the transitions that can occur next.

The transient mode is a way to account for the hybrid automation $HA$ dwell time requirement. This requirement guarantees that residuals, and hence consistency indicators, can be properly computed and that signature-events can be properly issued [Bayoudh et al., 2008]
5 Application Case Study

To illustrate the method, a part of the Barcelona sewer network presented in [Vento et al., 2011] is used (see Fig. 2). The elements that appear in the example are: two virtual tanks $T_0$ and $T_1$, a control gate (gate$_1$), two pluviometers $P_{19}$ and $P_{16}$ to measure the rain intensity and two liminometers $L_{30}$ and $L_{41}$ to measure the sewer level. The control gate is commanded by a controller applying open/close gate actions depending on the flow in the sewer.

$$\text{controller}$$

Figure 2: A small part of the sewer network

5.1 Hybrid Modeling

A hybrid automaton model can be obtained to represent the hybrid phenomena present in the network associated with the virtual tanks and the control gate. A way to obtain the hybrid model is to provide the automata for each component ($T_0$, $T_1$ and gate$_1$) and then synchronizing all automata to get the global model [Henzinger, 1996]. The automaton for a virtual tank is given by two discrete states: overflow ($O$) and non-overflow ($WO$) as is shown in Fig. 3. Regarding the control gate, there are four discrete states, the nominal behaviours (open or closed) and the faulty behaviours (stuck open or stuck closed).

(a) Automaton for a virtual tank (b) Automaton for a control gate

The global hybrid automaton has 16 operation modes where 8 of them correspond to nominal modes and the rest correspond to different configurations involving a control gate fault (stuck closed or stuck open). The set of structural faults is given by $F_S = \{f_1, f_2\}$ and the non-structural faults are used to model the faults in sensor ($L_{30}, L_{41}, P_{19}, P_{16}$) given by $F_{ns} = \{f_3, f_4, f_5, f_6\}$, respectively.

The set $\Sigma_s = \{\sigma_{o1}, \sigma_{o2}, \sigma_{u1}, \sigma_{u2}\}$ represents the unobservable spontaneous events. Event $\sigma_{u1}$ corresponds to the volume in tank $T_0$ reaching its maximum $v_{0\text{max}}$. Event $\sigma_{o2}$ corresponds to the input flow being less than the output flow from $T_0$ (i.e., $q_{0\text{in}} < q_{0\text{out}}$). The other events are related to the virtual tank $T_1$. The set $\Sigma_{fs} = \{\sigma_{f1}, \sigma_{f2}\}$ represents the fault events related to the structural faulty modes and $\Sigma_{fs} = \{\sigma_{f2}, \sigma_{f3}, \sigma_{f4}, \sigma_{f5}\}$ the non-structural fault events. The set $\Sigma = \{\sigma_{o1}, \sigma_{o2}\}$ gathers input events corresponding to closing or opening the valve issued by the controller.

5.2 Simulation Results

Assume that the system tracks the mode sequence $\{q_1, q_3, q_1, q_5\}$ and the sampling time is $\Delta t = 300s$. Mode $q_1$ refers to the situation in which no tank is in overflow. Then, $T_1$ is in overflow during a period of time (mode $q_3$) until it leaves the overflow situation (mode $q_1$). Later, the control gate is closed. The diagnoser must track the right mode sequence and detect and isolate the possible faults from an incrementally built behavior automaton $B$.

Assuming that the initial mode is known and it is $q_1$, then applying Algorithms 1 the initial $B$ is shown in Fig. 4. The initial diagnoser is obtained applying the propagation algorithm described in [Sampath et al., 1995] to the initial $B$. Then, the diagnoser waits for the occurrence of an event. Notice that the initial $B$ includes the possible events that may occur. These events are $\delta_{13}, \delta_{14}, \delta_{12}, \delta_{23}^*, \delta_{24}^*, \delta_{23}^*, \sigma_{o1}$ and $\sigma_{o2}$.

Figure 4: Initial incremental $B$

Notice that $Q_{disc} = Q_{o1} \cup Q_{o2} \cup Q_{u1} \cup Q_{u2} \cup Q_{fs}$. Fig. 5 shows the value of the set of residuals for all groups in $Q_{disc}$. According to the set of residuals for the set of modes of HA, two signature-events, $\delta_{14}$ and $\delta_{11}$, were identified using the consistency indicators appropriately. These signature-events correspond to transitions $q_1 \rightarrow q_3$ and...
$q_3 \rightarrow q_1$ with $q_1 \in Q_{q_3}$ and $q_3 \in Q_{q_3}$. Notice for instance that when the system is in mode $q_3$, $\Phi_{q_3}(k) \neq 0$ and $\Phi_{q_3}(k) = 0$. Both modes $q_1, q_3 \in Q$ represent a nominal behaviour.

Figure 5: Residuals generation for the set of modes in HA

Later, the observable event $e_{q_3}$ occurs, corresponding to the control gate closing. This event is identified instantaneously and indicates that a mode change from $q_1$ to $q_3$ takes place. Fig. 5 shows the set of residuals of mode $q_1$ and $q_3 \in Q_{q_3}$. It should be noticed that the residuals in mode $q_3$ are consistent with measurements after $\delta_{q_3}$ is detected, i.e. $\Phi_{q_3}(k) = 0$.

The set of residuals are only generated for modes that are visited in $HA$. In this way, the efficient use of memory is guaranteed. There is a set of two residuals per group using the expression given by (4).

A non-structural fault then occurs at 9000s (indicated in Fig. 5 with a black vertical dashed line). Then, the diagnoser detects the fault at 9300s. The set of consistency indicators of mode $q_3$ are used to isolate the fault. The observed signature is $[1 \ 0 \ 0]$ which, according to $FS_{q_3}$, corresponds to a fault in sensor $L_{10}$. Finally, the hybrid diagnoser stops and reports the diagnosis. Indeed, a non-structural faults needs to be repaired before the diagnoser can resume.

The report given by the hybrid diagnoser is shown in Table 1. The first column represents mode changes in HA, the second one, the identified events. The third column corresponds to the diagnoser state information and total number of states generated, the fourth one shows the total number of residuals generated. The last two columns show the occurrence time and the detection time of the identified events.

<table>
<thead>
<tr>
<th>Mode-change</th>
<th>Residuals events</th>
<th>Diagnosed residuals</th>
<th>Detection time (s)</th>
<th>Occurrence time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_1 \rightarrow q_3$</td>
<td>$r_1, r_2$</td>
<td>$r_1, r_2$</td>
<td>11</td>
<td>9300</td>
</tr>
</tbody>
</table>

Table 1: Hybrid diagnoser report

Table 2 provides a comparison of the results obtained with the present method and those obtained with the offline diagnoser generation [Vento et al., 2011; Bayoudh et al., 2008], standing out the benefits of the proposed method.

6 Conclusions

A method to incrementally build a hybrid diagnoser has been presented. The diagnoser is built whenever the system requires it after an event occurs (signature-event or input event). The method comprises the detection and isolation of structural and non-structural faults which are included in the system model. The diagnoser executes the tasks of mode recognition and identification using the consistency indicators generated from a set of residuals for every mode and then builds the part of the diagnoser required by the system operation. Thus, the diagnoser obtained requires less memory space and can be efficiently obtained online. An illustrative example of the proposed approach based on a piece of the Barcelona sewer network is used. Future work will consider to add the incremental design of the $HA$ from the component hybrid automata. This will nicely complete the proposed incremental approach, avoiding not only to store the whole diagnoser but also the whole hybrid model.

References


