A Framework for Active Fault-Tolerant Control of Deterministic I/O Automata

Melanie Schmidt * Jan Lunze *

* *Institute of Automation and Computer Control, Ruhr-Universität Bochum, Germany (e-mail: {schmidt, lunze}@atp.rub.de)*

Abstract: A method for the active fault-tolerant control (FTC) of systems modeled by deterministic input/output (I/O) automata is presented. In the fault-free case, a given controller moves the system such into a specified target state. The aim of the paper is to construct a framework which guarantees that the target state is reached again after the occurrence of a fault. For this, as usually in active FTC, two steps are performed when a fault is present: 1. Fault detection and identification, 2. Reconfiguration of the controller. In this paper, two existing methods for the fault diagnosis and the reconfiguration for I/O automata are combined in order to obtain an active and completely autonomous FTC framework. It is shown how the methods need to be modified and complemented in order to work conjointly and without any inadmissible assumptions on the results of other components. The applicability of the developed method is demonstrated by means of an example featuring a manufacturing cell.

Keywords: Fault tolerance, discrete-event systems, I/O automata, diagnosis, control design

1. INTRODUCTION

This paper deals with active fault-tolerant control (FTC) of discrete-event systems modeled by deterministic input/output (I/O) automata. As long as no fault is present, the plant $\mathcal{P}$ is controlled by the nominal controller $\mathcal{C}$ such that a given specification, for example the achievement of a final state, is fulfilled. When a fault occurs, the behavior of the plant is affected such that the nominal controller can not steer the plant according to the specification any longer. The aim of active FTC is to modify the controller $\mathcal{C}$ such that the faulty plant again adheres to the specification. After the occurrence of a fault, two main steps have to be performed by the active FTC method autonomously:

(1) Fault detection and identification
(2) Reconfiguration of the controller.

To accomplish these steps, an active FTC loop as shown in Fig. 1 is considered in this paper.

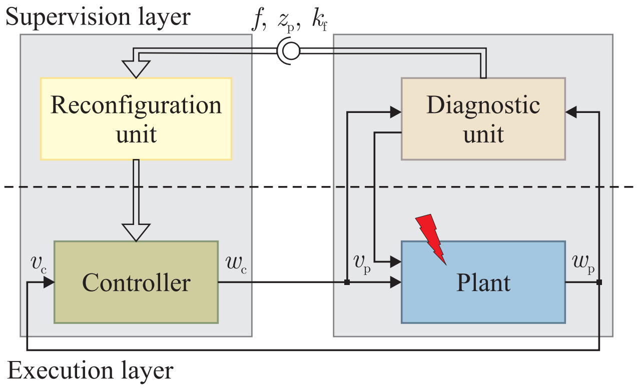

Fig. 1. Active FTC loop

The active FTC loop consists of two layers:

An execution layer, where the plant $\mathcal{P}$ and the controller $\mathcal{C}$ form a classical feedback control loop.

A supervision layer containing the diagnostic unit $\mathcal{D}$ and the reconfiguration unit $\mathcal{R}$.

In previous publications, both steps have been considered independently. In (Schmidt and Lunze (2013)), a method for the active diagnosis of deterministic I/O automata has been presented (cf. right part of Fig. 1), whereas (Nke and Lunze (2011b)) deals with the online reconfiguration for this system class (cf. left part of Fig. 1). Even though both methods consider the same system class, three main steps have to be undertaken in order to combine them to an active and completely autonomous FTC framework:

(1) The diagnostic unit $\mathcal{D}$ has to be complemented such as to provide exactly the information that the reconfiguration unit $\mathcal{R}$ requires.
(2) The existing methods have to be modified as some previously made assumptions (e.g. on the instantaneous identification of the fault) are no longer valid.
(3) A component managing the switching among inputs from the controller, the diagnostic unit and the reconfiguration unit has to be introduced.

Main contribution. The main contribution of the paper consists of combining the methods of active fault diagnosis (Schmidt and Lunze (2013)) and online reconfiguration (Nke and Lunze (2011b)) such that an active FTC framework for I/O automata is obtained. It is shown that the existing methods have to be complemented by a fault detection component, a method to determine the time at which the fault occurred as well as a mechanism coordinating the switching between nominal operation, fault diagnosis and reconfiguration.

Literature review. There are several approaches for the FTC of discrete event systems, most of which use standard automata and the Supervisory Control Theory (SCT) and fall into three categories:

(1) Robust or passive fault-tolerant controllers
(2) Switching controllers
(3) Controllers with checkpoint states.

To the best of our knowledge, no other approach for the fault-tolerant control of discrete event systems modeled by I/O automata as considered in this paper exists. This system class is especially suited for modeling dynamic systems in which the inputs from a controller and the outputs of the plant form an explicit causality relation. While the supervisors resulting from SCT are designed such as to *prevent* the system from executing forbidden motions, the control aim in this paper is to *explicitly steer* the system into a desired final state.

In the first category, a single controller is designed which guarantees the fulfillment of the nominal specification (Saboori and Hashtrudi Zad (2006), Park and Lim (1999)) or a degraded specification (Wen et al. (2008), Wittmann et al. (2012)) even when a fault occurred. Therefore, no explicit diagnostic result is necessary. In contrast to this, the method presented in this paper is an *active* FTC method for which the present fault has to be identified.

In the second category, a bank of controllers corresponding to different fault cases (Darabi et al. (2003)) or different system states at the time the fault occurs (Paoli et al. (2008)) is designed offline. Then, once a fault is identified, the respective controller is connected to the plant. On the contrary, in this paper only one nominal controller is designed, which is adapted online in case of a fault. Therefore, there is no need to precompute and store a controller for every possible fault.

In the third category, certain restart states within the controller are selected, to which the system is returned after a fault occurred and from which the process can be restarted afterwards (Andersson et al. (2009)). It is assumed that methods for the fault diagnosis and the physical transfer of the system into the restart state are given. In this paper, no explicit restart states are considered, but rather the system is always returned to its last correct state.

In summary the FTC method in this paper fundamentally differs from existing methods, because a different system class and control aim is considered and hence a completely different approach is necessary to solve the FTC problem.

Organization of the paper. In Sec. 2, the system model is introduced. The FTC problem to be solved is presented in Sec. 3. Sec. 4 contains the main result of the paper and describes the proposed FTC framework in detail. The developed method is demonstrated by applying it to an example featuring a manufacturing cell in Sec. 5.

2. SYSTEM MODEL

For each fault $f \in \mathcal{F}$ to be considered, a *deterministic I/O automaton*

$$\mathcal{A}_f = (\mathcal{Z}_f, \mathcal{V}, \mathcal{W}, G_f, H_f, z_{f0}) \tag{1}$$

with

- $\mathcal{Z}_f$ - state set, $\mathcal{V}$ - input set, $\mathcal{W}$ - output set
- $G_f : \mathcal{Z}_f \times \mathcal{V} \to \mathcal{Z}_f$ - state transition function
- $H_f : \mathcal{Z}_f \times \mathcal{V} \to \mathcal{W}$ - output function
- z_{f0} - initial state

is defined, which models the behavior of the system when the fault f is present. The state transition function

$$G_f(z(k), v(k)) = z(k+1) =: z'(k) \tag{2}$$

and the output function

$$H_f(z(k), v(k)) = w(k) \tag{3}$$

can be combined to the characteristic function $L_f : \mathcal{Z}_f \times \mathcal{W} \times \mathcal{Z}_f \times \mathcal{V} \to \{0, 1\}$, where

$$L_f(z', w, z, v) = \begin{cases} 1, & \text{if } G_f(z, v) = z' \wedge H_f(z, v) = w \\ 0, & \text{otherwise.} \end{cases}$$

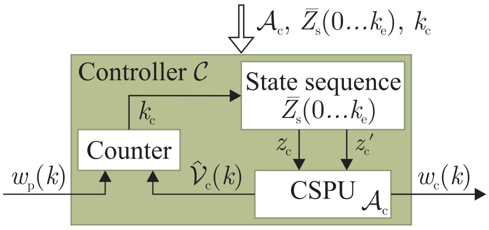
$\mathcal{A}_\mathrm{c}$, $\bar{Z}_\mathrm{s}(0\ldots k_\mathrm{e})$, k_c
Controller $\mathcal{C}$
State sequence
$\bar{Z}_\mathrm{s}(0\ldots k_\mathrm{e})$
k_c
Counter
z_c
z_c'
$w_\mathrm{p}(k)$
$\hat{\mathcal{V}}_\mathrm{c}(k)$
CSPU
$\mathcal{A}_\mathrm{c}$
$w_\mathrm{c}(k)$

Fig. 2. Realization of the feedback controller $\mathcal{C}$

The faultless system is modeled by a deterministic I/O automaton as well and is denoted by $\mathcal{A}_0$. Faults are assumed to change the state transition function G_f and the output function H_f compared to the faultless system $\mathcal{A}_0$, but neither its input set $\mathcal{V}$ nor its output set $\mathcal{W}$.

3. PROBLEM SETUP

3.1 Design of the controller

In this paper, the specification requires the cyclic achievement of a final state z_F. That is, it is aimed to find a controller $\mathcal{C}$ which steers the plant such that it reaches the state z_F starting from its initial state z_0 within a finite number of steps, returns to its initial state z_0 and the process starts over.

Problem 1. (Controller design). Given a plant $\mathcal{P}$ modeled by an automaton $\mathcal{A}_0$ and an initial state $z_\mathrm{p}(0) = z_0$, find a controller $\mathcal{C}$ which steers the plant according to

$$z_\mathrm{p}(\tilde{k}) = z_\mathrm{F} \tag{4}$$

$$z_\mathrm{p}(\tilde{k}+1) = z_0, \tag{5}$$

where $\tilde{k}$ is an arbitrary but finite time index.

A solution to this problem can be found in (Nke and Lunze (2011a)), where a controller $\mathcal{C}$ with the structure shown in Fig. 2 is proposed. Note that all elements related to the plant are labeled with the index “p”, while for the specification the index “s” and for the controller the index “c” is used.

The feedback controller $\mathcal{C}$ consists of three elements:

(1) The state sequence $\overline{Z}_\mathrm{s}(0 \ldots k_\mathrm{e}) = (z_\mathrm{s}(0), \ldots, z_\mathrm{s}(k_\mathrm{e}))$ to be followed by the plant, where k_e denotes the last element of a state sequence.