Conditions for Resolving Observability Problems in Distributed Testing

Jessica Chen[1], Robert M. Hierons[2], and Hasan Ural[3]

[1] School of Computer Science, University of Windsor
Windsor, Ontario, Canada N9B 3P4
xjchen@uwindsor.ca
[2] Department of Information Systems and Computing, Brunel University
Uxbridge, Middlesex, UB8 3PH United Kingdom
rob.hierons@brunel.ac.uk
[3] School of Information Technology and Engineering, University of Ottawa
Ottawa, Ontario, Canada K1N 6N5
ural@site.ottawa.ca

Abstract. Controllability and observability problems may manifest themselves during the application of a test or checking sequence in a test architecture where there are multiple remote testers. These problems often require the use of external coordination message exchanges among testers during testing. It is desired to construct a test or checking sequence from the specification of the system under test such that it will be free from these problems without requiring the use of external coordination messages. This paper investigates conditions that allow us to construct such a test or checking sequence. For specifications satisfying these conditions, procedures for constructing subsequences that eliminate the need for using external coordination messages are given.

1 Introduction

Testing an implementation of a system is often carried out by constructing an input sequence from the specification of the system, applying the input sequence in a test architecture, and analyzing the resulting output sequence to determine whether the implementation conforms to the specification on this input sequence. In distributed testing, a *distributed test architecture* is used where a tester is placed at each port of the *system under test* (SUT) N and an input sequence constructed from the specification M modeling the externally observable behaviour of N is applied. Such an input sequence is called a *test sequence* [11, 12] or *checking sequence* [4, 6] and is constructed from M to determine whether N is a correct or faulty implementation of M.

During the application of a test or checking sequence to N in a distributed test architecture, the existence of multiple testers brings out the possibility of coordination problems among remote testers known as *controllability* and *observability* problems. These problems occur if a tester cannot determine either when to apply a particular input to an SUT, or whether a particular output

D. de Frutos-Escrig and M. Núñez (Eds.): FORTE 2004, LNCS 3235, pp. 229–242, 2004.

from an SUT is generated in response to a specific input, respectively. Without loss of generality, let us consider a distributed architecture where there are two testers called, for instance, the lower tester (L) and the upper tester (U). In this architecture, U and L are two remote testers that are required to coordinate the application of a test or checking sequence. The controllability (synchronization) problem manifests itself when L (or U) is expected to send an input to N after N responds to an input from U (or L) with an output to U (or L), but L (or U) is unable to determine whether N sent that output. It is therefore important to construct a synchronizable test or checking sequence that causes no controllability problems during its application in the distributed test architecture. For some specifications, an input sequence can be constructed such that no two consecutive inputs will cause a controllability problem, and hence the coordination among testers is achieved indirectly through their interactions with N [12]. However, for some other specifications, there may not exist an input sequence in which the testers can coordinate solely via their interactions with N [1]. In this case it is necessary for testers to communicate directly by exchanging external coordination messages among themselves over a dedicated channel during the application of the input sequence [2].

During the application of even a synchronizable input sequence in a distributed test architecture, the observability problem manifests itself when L (or U) is expected to receive an output from N in response to either the previous input or the current input and because L (or U) is not the one to send the current input, L (or U) is unable to determine when to start and stop waiting. Such observability problems hamper the detectability of *output shift faults* in N i.e., an output associated with the current input is generated by N in response to either the previous input or the next input. To ensure the detectability of potential output shift faults in N the test or checking sequence needs to be augmented either by additional input subsequences selected from the specification M [10] or by external coordination message exchanges between testers [2] such that during the application of the input sequence testers can determine whether the output observed is received in response to the correct input as specified in M. Again, for some specifications, an input sequence can be constructed without using external coordination messages among testers such that no potential output shift faults will remain undetected, and hence the coordination among testers is achieved indirectly through their interactions with N. However, for some other specifications, there may not exist an input sequence in which observability problems can be resolved without using direct external coordination message exchanges among testers.

Both controllability problems and observability problems may be overcome through the use of external coordination messages. However, there is often a cost associated with the use of such messages. This cost includes the expense of implementing the infrastructure required in order to allow the messages to be sent and may also include a cost of a delay introduced by the sending of each message. It is thus desirable to construct a test or checking sequence from the specification of the system under test such that it will be free of controllability

and observability problems without requiring the use of external coordination message exchanges. Previous authors have investigated the problem of producing a test or checking sequence that either has no controllability problems or no controllability and observability problems and that either uses no external coordination message exchanges or uses a minimum number of external coordination message exchanges (see, for example, [1, 3, 5, 7, 8, 9, 13, 14, 15, 16]).

This paper investigates conditions that allow us to construct a test or checking sequence without encountering controllability and observability problems and without using external coordination messages among testers. The rest of the paper is organized as follows: Section 2 introduces the terminology. Section 3 gives a formal definition of the problem and identifies the conditions that the specification of the system under test is checked against. Section 4 presents new procedures for constructing subsequences that eliminate the need for using external coordination messages: for a transition t and port p we produce a subsequence that does not allow a fault in the output of t at p to be masked by an output shift fault. Section 5 gives the concluding remarks.

2 Preliminaries

2.1 FSM and Its Graphical Representation

An *n-port Finite State Machine* M (simply called FSM M below) is defined as $M = (S, I, O, \delta, \lambda, s_0)$ where

- S is a finite set of states of M;
- $s_0 \in S$ is the initial state of M;
- $I = \bigcup_{i=1}^{n} I_i$, where I_i is the input alphabet of port i, and $I_i \cap I_j = \emptyset$ for $i, j \in [1, n]$, $i \neq j$;
- $O = \prod_{i=1}^{n}(O_i \cup \{-\})$, where O_i is the output alphabet of port i, and $-$ means null output;
- δ is the transition function that maps $S \times I$ to S, i.e., $\delta : S \times I \rightarrow S$;
- λ is the output function that maps $S \times I$ to O, i.e., $\lambda : S \times I \rightarrow O$.

Note that each $y \in O$ is a *vector of outputs*, i.e., $y =< o_1, o_2, ..., o_n >$ where $o_i \in O_i \cup \{-\}$ for $i \in [1, n]$. We will use $*$ to denote any possible output, including $-$, at a port. We also use $*$ to denote any possible input or any possible vector of outputs of a transition. In the following, $p \in [1, n]$ is a port, $x \in I$ is a general input, and $x^p \in I_p$ is an input at specific port p. We use $y[p, c]$ to denote the output vector y of a transition whose output at port p is c. Alternatively, we use $y \mid_p$ to denote the output at port p in y.

A *transition* of an FSM M is a triple $t = (s_1, s_2, x/y)$, where $s_1, s_2 \in S$, $x \in I$, and $y \in O$ such that $\delta(s_1, x) = s_2$, $\lambda(s_1, x) = y$. s_1 and s_2 are called the *starting state* and the *ending state* of t respectively. The *input/output pair* x/y is called the *label* of the transition. A transition $(s_1, s_2, x/y)$ will also be denoted as $s_1 \xrightarrow{x/y} s_2$.

A *path* $\rho = t_1\ t_2\ \ldots\ t_k$ $(k \geq 0)$ is a finite sequence of transitions such that for $k \geq 2$, the ending state of t_i is the starting state of t_{i+1} for all $i \in [1, k-1]$. When the ending state of the last transition of path ρ_1 is the starting state of the first transition of path ρ_2, we use $\rho_1@\rho_2$ to denote the *concatenation* of paths ρ_1 and ρ_2. The *label* of a path $(s_1, s_2, x_1/y_1)\ (s_2, s_3, x_2/y_2)\ \ldots\ (s_k, s_{k+1}, x_k/y_k)$ $(k \geq 1)$ is the sequence of input/output pairs $x_1/y_1\ x_2/y_2\ \ldots\ x_k/y_k$ which is called an *input/output sequence*. We will consider FSMs that are free from *same-port-output-cycles* and *isolated-port-cycles*. A same-port-output-cycle in an FSM is a path $(s_1, s_2, x_1/y_1)\ (s_2, s_3, x_2/y_2) \ldots (s_k, s_{k+1}, x_k/y_k)$ $(k \geq 2)$ such that $s_1 = s_{k+1}$, $s_i \neq s_{i+1}$ for $i \in [1, k]$, and there exists a port p with $y_i\ |_p \neq -$ and $x_i \notin I_p$ for all $i \in [1, k]$. An isolated-port-cycle in an FSM is a path $(s_1, s_2, x_1/y_1)$ $(s_2, s_3, x_2/y_2)\ \ldots\ (s_k, s_{k+1}, x_k/y_k)$ $(k \geq 2)$ such that $s_1 = s_{k+1}$, $s_i \neq s_{i+1}$ for $i \in [1, k]$, and there exists a port p with $y_i\ |_p = -$ and $x_i \notin I_p$ for all $i \in [1, k]$.

We will use 2-port FSMs to show some examples. In a 2-port FSM, we will denote the ports U and L to stand for the upper interface and the lower interface of the FSM. We use u, u_1, u_2, ... to denote inputs at port U, and l, l_1, l_2, ... to denote the inputs at port L. The output vector $y = \langle o_1, o_2 \rangle$ on the label of a transition of the 2-port FSM are pairs of output $o_1 \in O_1$ at port U and output $o_2 \in O_2$ at port L.

2.2 Controllability (Synchronization) Problem

Given an FSM M and an input/output sequence $x_1/y_1\ x_2/y_2\ \ldots\ x_k/y_k$ of M, where $x_i \in I$ and $y_i \in O$, $i \in [1, k]$, a *controllability (synchronization) problem* occurs when, in the labels x_i/y_i and x_{i+1}/y_{i+1} of any two consecutive transitions, $\exists p \in [1, n]$ such that $x_{i+1} \in I_p$, $x_i \notin I_p$, $y_i\ |_p = -$ $(i \in [1, k-1])$. Two consecutive transitions t_i and t_{i+1} whose labels are x_i/y_i and x_{i+1}/y_{i+1}, form a *synchronizable pair* of transitions if t_{i+1} can follow t_i without causing a synchronization problem. Any (sub)sequence of transitions in which every pair of transitions is synchronizable is called a *synchronizable transition (sub)sequence*. An input/output sequence is said to be *synchronizable* if it is the label of a synchronizable transition sequence.

2.3 Observability Problem

Suppose we are given an FSM M and an input/output sequence $x_1/y_1\ x_2/y_2$ $\ldots\ x_k/y_k$ of M, where $x_i \in I$ and $y_i \in O$, $i \in [1, k]$. A *1-shift output fault* in an implementation N of M exists when, in the labels x_i/y_i and x_{i+1}/y_{i+1} of any two consecutive transitions, one of the following holds:

1. There exists $o \in O_p$ and $p \in [1, n]$ such that $y_i\ |_p = o$, $y_{i+1}\ |_p = -$, N produces output $-$ at p in response to x_i after $x_1 \ldots x_{i-1}$, and N produces output o at p in response to x_{i+1} after $x_1 \ldots x_i$.
2. There exists $o \in O_p$ and $p \in [1, n]$ such that $y_i\ |_p = -$, $y_{i+1}\ |_p = o$, N produces output o at p in response to x_i after $x_1 \ldots x_{i-1}$, and N produces output $-$ at p in response to x_{i+1} after $x_1 \ldots x_i$.

An instance of the observability problem manifests itself as a *potentially undetectable 1-shift output fault* if there is a 1-shift output fault related to $o \in O_p$ in any two consecutive transitions whose labels are x_i/y_i and x_{i+1}/y_{i+1}, such that $x_{i+1} \notin I_p$. In this case, we say that the tester at port p is *involved* in the shift, and would not be able to detect it.

3 Problem Definition and Conditions

3.1 Problem Definition

Suppose that a specification of the system under test is given as an n-port FSM. Each potential undetectable 1-shift output fault in the given FSM is related to a pair of transitions t_1 and t_2 adjacent at a state of the FSM. Clearly, one can identify all potential undetectable 1-shift output faults in the given FSM using the definition in Section 2.3, and form a set of pairs of transitions where each pair of transitions t_1 and t_2 is related to a potential undetectable shift of an output at a specific port p. Note that, if there exists a potential undetectable forward shift of output d at port p from transition t_1 to transition t_2, then, this potential shift may be realized in an implementation, by a missing output d at port p in t_1 and an extra output d at port p in t_2. Similarly, if there exists a potential undetectable backward shift of output d at port p from transition t_2 to transition t_1, then, this potential shift may be realized in an implementation, by an extra output d at port p in t_1 and a missing output d at port p in t_2.

Therefore, one has to determine, for each pair of transitions t_1 and t_2 related to a potential undetectable shift of an output at a specific port p, whether there exists a missing/extra output at port p of transition t, for each transition t in t_1t_2. In order to do this without using external coordination messages among testers, one has to determine whether there is a subsequence of transitions in the given FSM that will detect a missing/extra output at port p of transition t and then construct such a subsequence. Clearly, we also need to check whether every incoming transition of each state of the FSM forms a synchronizable pair of transitions with at least one of the outgoing transitions of that state and every outgoing transition of each state of the FSM forms a synchronizable pair of transitions with at least one of the incoming transitions of that state. If this condition does not hold, then the FSM is called *intrinsically non-synchronizable*. In this paper, we consider FSMs that are not intrinsically non-synchronizable.

Hence, the problem we consider is the following: for each transition t in each pair of transitions t_1 and t_2 of the FSM related to a potential undetectable shift of an output at a specific port p, we wish to produce a single subsequence $\rho_1@t@\rho_2$ that checks whether the output produced by t at p represents a missing/extra output at port p. In addition, the subsequence $\rho_1@t@\rho_2$ must have the following properties: it must be synchronized and it cannot contain an undetectable output shift fault involving t at port p. This places conditions on ρ_1 (called the leading path) and ρ_2 (called the trailing path). We give conditions under which such ρ_1 and ρ_2 exists. We also give algorithms that, when these conditions hold, generate ρ_1 and ρ_2 for given t and p. Where such subsequences exist for *both* t_1 and t_2

we say that they *resolve* the potential undetectable shift of the output at port p in t_1t_2. In the rest of the paper, we will not qualify a subsequence or sequence as "synchronizable" as only synchronizable sequences and subsequences will be considered.

3.2 Conditions

Suppose that two transitions in the given FSM may be sequenced in a manner that gives a potential undetectable shift of an output at port p. The following gives conditions under which these two transitions may be checked separately for a missing/extra output at p, without using external coordination messages, in a manner that does not allow them to be involved in an undetectable shift of an output at p. Further, it will transpire that under these conditions, this may be achieved for *every* pair t_1t_2 of transitions from the FSM. In Section 5 we will discuss how these conditions may be weakened when we are considering a given test or checking sequence derived from the given FSM.

Given an FSM with no same-port-output-cycles or isolated-port-cycles, we can resolve all of its potential undetectable 1-shift output faults without using external coordination messages if and only if for any pair of transitions $s_1 \xrightarrow{*/*} s$ *and* $s \xrightarrow{*/*} t_1$ *in the FSM,*

a *if there exists a potential undetectable forward shift of an output at port p, then there exists at least one transition to s with a null output at port p, and at least one transition from s with either an input or a non-empty output at port p.*

b *if there exists a potential undetectable backward shift of an output at port p, then there exists at least one transition to s with a non-empty output at port p, and at least one transition from s with either an input or a null output at port p.*

Below we show that the condition for the case of potential undetectable *forward* shift (i.e. part a.) is *necessary*. The necessity of the condition for the case of potential undetectable *backward* shift is analogous. In Section 4 we will show how, under these conditions, the potential undetectable output shift faults may be resolved without the use of external coordination messages and thus that these conditions are *sufficient* conditions.

Suppose we have transitions $t_1 = s_1 \xrightarrow{*/y_1[p,d]} s$ and $t_2 = s \xrightarrow{x/y_2[p,-]} r_1$ in the FSM of the specification where $x \notin I_p$. Thus, there is a potential undetectable forward shift of output d from t_1 to t_2.

– Suppose that the condition does not hold, and $\forall s'$ such that $s' \xrightarrow{x'/y'} s$, we have $y' \mid_p \neq -$. In this situation, the output at port p on any transition leading to s may be shifted to transition t_2, and these potential shifts are undetectable. So we have no way to check that transition t_2 has null output at port p in the implementation, because no matter how we get to state s,

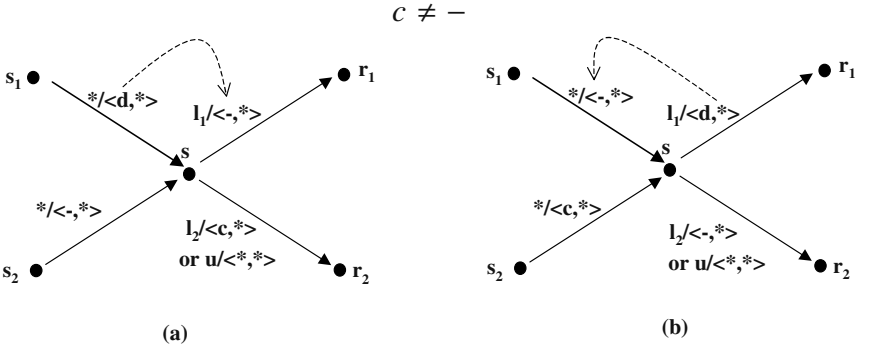

Fig. 1. illustration of the condition

there is always a possibility of an undetectable forward output shift at port p to t_2.

– Suppose that the condition does not hold, and $\forall s'$ such that $s \xrightarrow{x'/y'} s'$, we have $x' \notin I_p$ and $y' \mid_p = -$. In this situation, the output d on transition t_1 may be shifted to any transition starting from s, and these potential shifts are undetectable. So we have no way to check that transition t_1 has output d at port p in the implementation, because no matter how we continue from state s, there is always a possibility of an undetectable forward shift of output d from t_1.

Figure 1 illustrates an example of a 2-port FSM where the condition is satisfied. In this figure, we have potential undetectable forward output shift fault in (a) and potential undetectable backward output shift fault in (b), as the dashed arrows show. For such potential faults, we have transition from s_2 to s and transition from s to r_2, so the condition holds. In fact, in (a), transition from s to r_2 will be used to check if there is a missing output d in transition $s_1 \xrightarrow{*/<d,*>} s$, as a result of a forward output shift. The transition from s_2 to s will be used to check if there is an extra output in transition $s \xrightarrow{l_1/<-,*>} r_1$ as a result of a forward output shift. The transitions from s_2 to s and from s to r_2 in (b) will be used analogously.

Note that

– In $s_1 \xrightarrow{*/y_1[p,d]} s \xrightarrow{x/y_2[p,-]} r_1$ $(x \notin I_p)$, having checked that the first transition does not have a missing output d at port p cannot guarantee that the second transition does not have an extra output at port p. In Figure 2 (a), suppose that using $s_2 \xrightarrow{*/<-,*>} s \xrightarrow{l_1/<-,*>} r_1$ we know that in transition $s \xrightarrow{l_1/<-,*>} r_1$ there is no extra output. However, we cannot jump to the conclusion that in $s_1 \xrightarrow{*/<d,*>} s$ there is no missing output d,

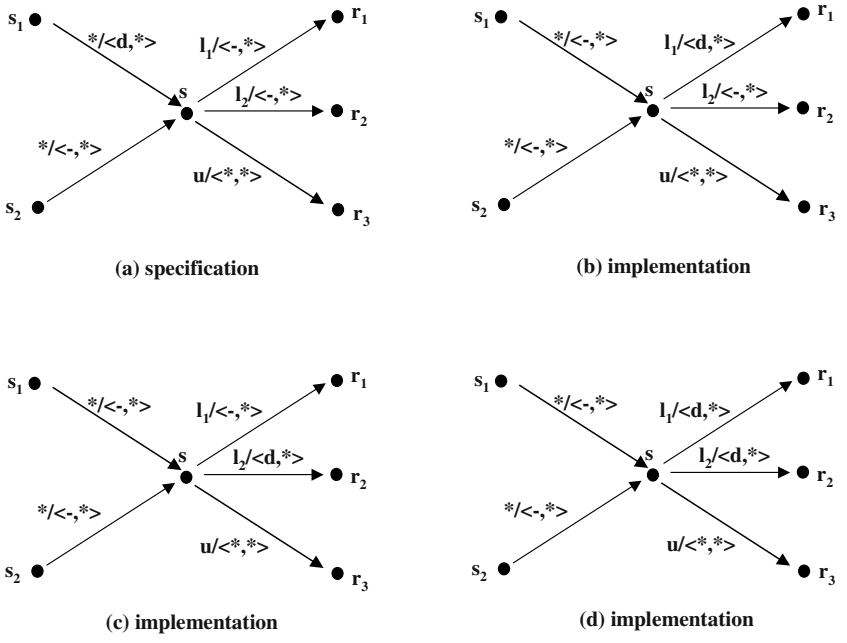

Fig. 2. An example to show possible undetectable 1-shift output faults

because Figure 2 (c) can be an implementation: Thus, we need to check both transition $s_1 \xrightarrow{*/<d,*>} s$ for possible missing output d, and transition $s \xrightarrow{l_1/<-,*>} r_1$ for possible extra output.

– The potential undetectable output shift faults may not be paired. In Figure 2, (a) may be implemented as in (d). Thus, we need to check transition $s_1 \xrightarrow{*/<d,*>} s$ for possible missing output d, and check both transitions $s \xrightarrow{l_1/<-,*>} r_1$ and $s \xrightarrow{l_2/<-,*>} r_2$ for possible extra output.

When the above condition holds, given a transition t, we show below how to check if there is a missing output or an extra output at a specific port on this transition in the implementation. To do so, we construct a *leading path* ρ_1 that leads to the starting state of t and a *trailing path* ρ_2 that starts from the ending state of t so that by applying subsequence $\rho_1@t@\rho_2$ to the implementation, we can detect if there is a missing/extra output at a specific port in transition t.

Note that we consider FSM with no same-port-output-cycles and no isolated-port-cycles. In this setting, our procedures to construct the leading paths and trailing paths will always terminate with subsequences that are adequate.

In Section 5 we will discuss the case where a test or checking sequence ρ has been given and we wish to produce subsequences to check for any 1-shift output faults that are undetectable in ρ.

4 Generating Subsequences

4.1 Checking the Potential Missing Data Output

Given a transition $t = s_1 \xrightarrow{*/y[p,d]} s_2$ $(d \neq -)$, in order to check that there is no missing output d at port p in this transition in the implementation, we construct a leading path to s_1 such that (i) it starts with a transition that has an input at port p; (ii) except for the first one, there is no other transition along the path that has an input at port p; (iii) on any transition of the path, there is no possibility of an extra output d at port p.

The leading path should start with a transition which can be used to help identify the potential missing output d in transition t. Obviously, such a transition should contain input or output at port p. Note that the leading path may be preceded in testing by other transitions that are capable of producing extra output at p. Thus it is not sufficient to define a leading path starting with a transition that has non-empty output at port p: We require that the leading path start with an *input* at port p. Furthermore, to minimize the length of the leading path, we require that the input at port p at the beginning is the only input at port p in the leading path. We require that along the leading path there is no possibility of an extra output d at port p, because the occurrence of such an extra output d will prevent us from detecting the missing output d at port p in transition t.

The following procedure constructs the leading path ρ_1.

i Let $\rho_1 = \varepsilon$
ii If the transition t has input $x^p \in I_p$, terminate as no leading path is required.
iii Let $v = s_1$
iv If $\exists v'.\ v' \xrightarrow{x^p/y'[p,c]} v$ and $c \neq -$, then $\rho_1 = (v', v; x^p/y'[p,c])@\rho_1$, terminate.
Otherwise, let v' be a state such that $v' \xrightarrow{x/y'[p,c]} v$ and $c \neq -$. Let $\rho_1 = (v', v; x/y'[p,c])@\rho_1$, $v = v'$, repeat iv.

For the first time in step (iv), if $\exists v''.\ v'' \xrightarrow{*/y'[p,-]} s_1$, then there exists a potential undetectable backward shift of output d between $v'' \xrightarrow{*/y'[p,-]} s_1$ and $s_1 \xrightarrow{*/y[p,d]} s_2$. According to our condition b., $\exists v'.\ v' \xrightarrow{*/y'[p,c]} s_1$ and $c \neq -$. The same argument holds for the repeated step (iv). Thus, in (iv), the existence of v' is guaranteed. Termination, with an adequate leading path, is guaranteed because M must be free from same-port-output-cycles. Further, since M has no same-port-output-cycles this procedure cannot repeat a state of M and thus, if M has m states, the leading path can have length at most $m - 1$.

At the end of the procedure, we have a leading path that, if it is non-empty, starts from a transition with an input at port p, followed by transitions with non-empty outputs at port p. Any of these outputs at p can be missing, but there is *no possibility* of an extra output at p.

Note that these outputs at port p can be d, and so, although there is no possibility of an extra d at port p, there exists possibility of a *missing* d in the leading path. So if a missing d at port p is detected, we may not be able to tell which d is missing: from a transition in the leading path or from transition $s_1 \xrightarrow{*/y[p,d]} s_2$. Additional conditions might be added to avoid this from happening. An example of such a condition is: *no two consecutive transitions have the same output data at the same port.*

The construction of the trailing path is quite similar: Given transition $s_1 \xrightarrow{*/y[p,d]} s_2$ ($d \neq -$), in order to check that there is no missing d in this transition in the implementation, we construct a trailing path starting from s_2 such that (i) it ends with a transition that has an input at port p; (ii) except for the last one, there is no other transition along the path that has an input at port p; (iii) on any transition of the path, there is no possibility of an extra output d at port p.

The following procedure constructs the trailing path ρ_2.

i Let $\rho_2 = \varepsilon$, $v = s_2$

ii If $\exists v'.\ v \xrightarrow{x^p/y'} v'$ for some input $x_p \in I_p$, then $\rho_2 = \rho_2 @ (v, v'; x^p/y')$, terminate.

Otherwise, let v' be a state such that $v \xrightarrow{x/y'[p,c]} v'$ and $c \neq -$. Let $\rho_2 = \rho_2 @ (v, v'; x/y'[p, c])$, $v = v'$, repeat ii.

Again, this must terminate, with an adequate trailing path of length at most $m - 1$.

4.2 Checking the Potential Extra Data Output

Given transition $s_1 \xrightarrow{*/y[p,-]} s_2$, in order to check that there is no extra output at port p in this transition in the implementation, we construct a leading path to s_1 such that (i) it starts from a transition with an input at port p; (ii) except for the first one, there is no other transition along the path that has an input at port p; (iii) on any transition of the path, there is no possibility of a missing output at port p.

The following procedure constructs the leading path ρ_1.

i Let $\rho_1 = \varepsilon$

ii If the transition t has input $x^p \in I_p$, terminate as no leading path is required.

iii Let $v = s_1$

iv If $\exists v'.\ v' \xrightarrow{x^p/y'[p,-]} v$, let $\rho_1 = (v', v; x^p/y'[p, -]) @ \rho_1$, terminate.

Otherwise, let v' be a state s.t. $v' \xrightarrow{x/y'[p,-]} v$. Let $\rho_1 = (v', v; x/y'[p, -]) @ \rho_1$, $v = v'$, repeat iv.

For the first time in step (iv), if $\exists v''.\ v'' \xrightarrow{*/y'[p,c]} s_1$ where $c \neq -$, then there exists potential undetectable forward shift of output c between $v'' \xrightarrow{*/y'[p,c]} s_1$ and $s_1 \xrightarrow{*/y[p,-]} s_2$. According to our condition a., $\exists v'.\ v' \xrightarrow{*/y'[p,-]} s_1$. The same argument holds for the repeated step (iv). Thus, in (iv), the existence of v' is guaranteed. Termination , with an adequate leading path of length at most $m-1$, is guaranteed because M must be free from isolated-port-cycles.

The leading path, if it is non-empty, starts from a transition with an input at port p, followed by a sequence of transitions with empty output at port p, and so there is no possibility of missing output at port p along the leading path.

The construction of the trailing path is similar: Given transition $s_1 \xrightarrow{*/y[p,-]} s_2$, in order to check that there is no extra output at port p in this transition in the implementation, we construct a trailing path from s_2 such that (i) it ends with a transition with an input at port p; (ii) except for the last one, there is no other transition along the path that has an input at port p; (iii) on any transition of the path, there is no possibility of a missing output at port p.

The following procedure constructs the trailing path ρ_2.

i Let $\rho_2 = \varepsilon$, $v = s_2$

ii If $\exists v'.\ v \xrightarrow{x^p/y'} v'$ for some $x^p \in I_p$, let $\rho_2 = \rho_2 @ (v, v'; x^p/y')$, terminate.
Otherwise, let v' be a state s.t. $v \xrightarrow{x/y'[p,-]} v'$. Let $\rho_2 = \rho_2 @ (v, v'; x/y'[p,-])$, $v = v'$, repeat ii.

Clearly, this must terminate, with an adequate trailing path of length at most $m-1$.

5 Concluding Remarks

Where a test architecture has remote testers it is necessary to consider controllability and observability issues. These problems often require the use of external coordination message exchanges among testers during testing. However, there is often a cost associated with the use of such messages: the cost of implementing the infrastructure required in order to allow the messages to be sent and possibly also a cost (or delay) due to the sending of each message. It is thus desirable to construct a test or checking sequence from the specification of the system under test such that it will be free of controllability and observability problems without requiring the use of external coordination message exchanges.

This paper investigates conditions that must be satisfied by the specification of the system under test for us to be able to produce a test for each transitions such that the test is free from controllability and observability problems. This problem is represented in the following way.

For each potential undetectable 1-shift output fault in the FSM, we have a pair of transitions $t_1 t_2$. For each transition t in $t_1 t_2$, we wish to produce

a single subsequence $\rho_1@t@\rho_2$ that checks the output produced by transition t at port p. It is necessary to precede t by some appropriate sequence as the starting state of t must be reached in order to execute t and the sequence used to reach this state must not be able to lead to a potentially undetectable shift of the same output involving t. It is necessary to follow t with some appropriate sequence since we must ensure that the sequence following t does not allow a potentially undetectable shift of the same output involving t[1]. The effectiveness of the subsequence $\rho_1@t@\rho_2$, at checking the output of t at p, must not be affected by controllability and observability problems. This paper gives necessary and sufficient conditions for there to be such a subsequence for each t in a pair of transitions t_1t_2 representing a potential undetectable output shift fault related to an output at port p. Further, given a transition t and port p, we have given algorithms that (if these conditions hold) produce a subsequence that checks the output of t at p without suffering from controllability and observability problems.

In practice, weaker conditions than those given in this paper will often suffice since:

i it may not be necessary to consider all pairs of transitions; and
ii it may be possible to use more than one subsequence to check a transition t.

We will now briefly discuss these factors. First, we may be concerned with potential observability problems in a *given* test or checking sequence ρ. Since some transition pairs representing potentially undetectable output shift faults may not be in ρ and since the paths leading to and trailing from those t_1t_2 pairs in ρ representing potential undetectable output shift faults are already determined in ρ, weaker conditions may suffice.

Suppose that ρ contains the subsequence t_1t_2 for transitions t_1 and t_2 and that these can participate in a potentially undetectable 1-shift output fault at port p. Sometimes we can eliminate this potentially undetectable 1-shift output fault by using a subsequence that checks the output of t_1 at p *or* a subsequence that checks the output of t_2 at p but not both. We now explain why this is the case. Suppose that we test the output of t_1 at p and find this to be correct; the case where we have checked the output of t_2 at p is similar. If we know that the subsequence t_1t_2 produces the (overall) correct output at p then we also know that t_2 produces the expected output at p. Naturally, further conditions must be placed on ρ in order for it to determine the overall output of t_1t_2 at p in the SUT.

This paper has investigated conditions under which it is possible to resolve potentially undetectable output shift faults. However, a number of questions remain. As already noted, weaker conditions sometimes suffice - the challenge is to produce general necessary and sufficient conditions. Suppose we produce a subsequence for each transition/port combination and generate a single sequence ρ that contains each of these subsequences. Then ρ is guaranteed to determine

[1] Sometimes it is possible to separate several test or checking sequences using a reset. Where this is the case, under certain conditions it is possible to replace ρ_2 with a reset.

correctness under the fault model in which only output faults are possible. However, this need not be an efficient sequence for this fault model. There is also the problem of producing an efficient test or checking sequence, from an FSM, that is guaranteed to determine correctness for more general fault models.

Acknowledgements

This work was supported in part by Natural Sciences and Engineering Research Council (NSERC) of Canada under grants OGP 976 and 209774, Leverhulme Trust grant number F/00275/D, Testing State Based Systems, and Engineering and Physical Sciences Research Council grant number GR/R43150, Formal Methods and Testing (FORTEST).

References

[1] S. Boyd and H. Ural. The synchronization problem in protocol testing and its complexity. *Information Processing Letters*, 40:131–136, 1991. 230, 231

[2] L. Cacciari and O. Rafiq. Controllability and observability in distributed testing. *Information and Software Technology*, 41:767–780, 1999. 230

[3] W. Chen and H. Ural. Synchronizable checking sequences based on multiple UIO sequences. *IEEE/ACM Transactions on Networking*, 3:152–157, 1995. 231

[4] A. Gill. *Introduction to the Theory of Finite-State Machines*. New York: McGraw-Hill, 1962. 229

[5] S. Guyot and H. Ural. Synchronizable checking sequences based on UIO sequences. In *Proc. of IFIP IWPTS'95*, pages 395–407, Evry, France, September 1995. 231

[6] F. Hennie. Fault detecting experiments for sequential circuits. In *Proc. of Fifth Ann. Symp. Switching Circuit Theory and Logical Design*, pages 95–110, Princeton, N. J., 1964. 229

[7] R. M. Hierons. Testing a distributed system: generating minimal synchronised test sequences that detect output-shifting faults. *Information and Software Technology*, 43(9):551–560, 2001. 231

[8] R. M. Hierons and H. Ural. UIO sequence based checking sequences for distributed test architectures. *Information and Software Technology*, 45(12):793–803, 2003. 231

[9] G. Luo, R. Dssouli, and G. v. Bochmann. Generating synchronizable test sequences based on finite state machine with distributed ports. In *The 6th IFIP Workshop on Protocol Test Systems*, pages 139–153. Elsevier (North-Holland), 1993. 231

[10] G. Luo, R. Dssouli, G. v. Bochmann, P. Venkataram, and A. Ghedamsi. Test generation with respect to distributed interfaces. *Computer Standards and Interfaces*, 16:119–132, 1994. 230

[11] K. Sabnani and A. Dahbura. A protocol test generation procedure. *Computer Networks*, 15:285–297, 1988. 229

[12] B. Sarikaya and G. v. Bochmann. Synchronization and specification issues in protocol testing. *IEEE Transactions on Communications*, 32:389–395, April 1984. 229, 230

[13] K. Tai and Y. Young. Synchronizable test sequences of finite state machines. *Computer Networks*, 13:1111–1134, 1998. 231

[14] H. Ural and Z. Wang. Synchronizable test sequence generation using UIO sequences. *Computer Communications*, 16:653–661, 1993. 231
[15] D. Whittier. Solutions to controllability and observability problems in distributed testing. Master's thesis, University of Ottawa, Canada, 2001. 231
[16] Y. Young and K. Tai. Observation inaccuracy in conformance testing with multiple testers. In *Proc. of IEEE WASET*, pages 80–85, 1998. 231