Overcoming controllability problems with fewest channels between testers

R. M. Hierons [a] H. Ural [b]

[a] *School of Information Systems, and Computing Mathematics, Brunel University, Uxbridge, Middlesex, UB8 3PH*

[b] *School of Information Technology and Engineering, University of Ottawa Ottawa, Ontario, Canada K1N 6N5*

Abstract

When testing a system that has multiple physically distributed ports/interfaces it is normal to place a tester at each port. Each tester observes only the events at its port and it is known that this can lead to additional controllability problems. While such controllability problems can be overcome by the exchange of external coordination messages between the testers, this requires the deployment of an external network and may thus increase the costs of testing. The problem studied in this paper is finding a minimum number of coordination channels to overcome controllability problems in distributed testing. Three instances of this problem are considered. The first problem is to find a minimum number of channels between testers in order to overcome the controllability problems in a given test sequence to be applied in testing. The second problem is finding a minimal set of channels that allow us to overcome controllability problems in any test sequence that may be selected from the specification of the system under test. The last problem is to find a test sequence that achieves a particular test objective and in doing so allows fewest channels to be used.

Key words: distributed test architecture, controllability problem, minimizing channels, test execution.

1 Introduction

In distributed testing, a *distributed test architecture* is used to test a system under test (SUT) N where a tester is placed at each port of N and a test sequence, derived from the specification of N, is applied. The application of the test sequence in this architecture requires the coordination amongst the remote testers. This requirement may lead to *controllability* and *observability*

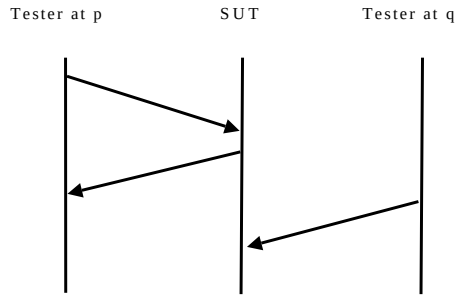

Fig. 1. A controllability problem

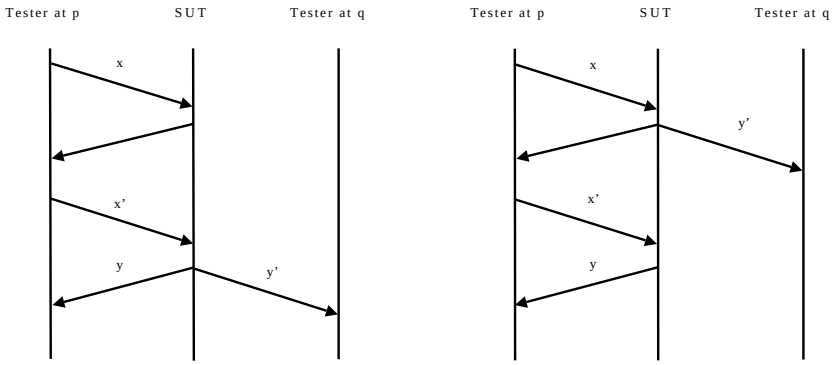

Fig. 2. An observability problem

problems due to the absence of a global clock. These problems occur if a tester cannot determine either when to apply a particular input to N, or whether a particular output from N is generated in response to a specific input, respectively. That is, the *controllability (synchronisation)* problem occurs when the tester at a port q is expected to send an input to N after N responds to an input from the tester at some port $p \neq q$, without sending an output to q and so the tester at port q is not able to determine when to send its input. This is illustrated in Figure 1.

Let us suppose, for example, that the input of x at port p is expected to lead to output y at p only and this should be followed by input x' at a port $q \neq p$. Then the tester at q does not observe either x or y and so cannot know when to send x'. The *observability problem* occurs when a tester at a port j is expecting to receive an output from N in response to either the previous input or the current input and it is not the tester to send the current input and so the tester at port j is not able to determine which input caused the output. Let us suppose, for example, that the input of x at port p is expected to lead to output y at p only, this is to be followed by input x' at p and this should lead to output of y at p and y' at $q \neq p$. Then the tester at p expects to observe $xyx'y$ and the tester at q expects to observe y'. This is still the case if the SUT responds to the input of x by producing y at p and y' at q and responds to input x' with output of y at p only. This is illustrated in Figure 2.

This paper concentrates on the situation in which the required behaviour of the SUT is represented by a *Finite State Machine (FSM)* that has multiple ports and black-box testing is being applied. Previous work on testing in the distributed test architecture has focussed on testing from an FSM since FSMs are suitable for modelling state-based systems and can be easily adapted to the case where there are multiple ports (see, for example, [6,7,13–15]). However, some of the results simply concern the problem of applying a given test sequence and thus do not depend on the use of an FSM.

Controllability and observability problems can be avoided if some necessary and sufficient conditions hold [2,3]. However, due to the restrictive nature of these conditions, controllability and observability problems often cannot be avoided and there is then the need to overcome these problems via coordination among remote testers during the application of a given test sequence. The coordination among testers is facilitated by using (external) coordination channels through which the testers can send (external) coordination messages (see, for example, [1,16]). For instance, to resolve the above controllability problem, a coordination channel from the tester at port i to the tester at port j is introduced through which the tester at port i, by sending a coordination message, notifies the tester at port j that it should send its input. In most cases, the use of coordination message exchanges to overcome observability problems in a given test sequence can be avoided by appending additional test subsequences to the test sequence [15]. Therefore, we will focus only on the controllability problem in the rest of the paper.

The use of coordination messages can lead to delays in testing and there has thus been much interest in minimizing the number of such coordination messages in order to overcome any controllability problems that may occur (see, for example, [4,9,14,15,17]). However, there is a cost associated with introducing coordination channels through which coordination messages can be sent and this cost depends on the number of channels, between the testers, that are required. The cost of setting up coordination channels can be more important than that of sending coordination messages and once a channel is set up, one could use it for exchanging as many coordination messages as necessary rather than incurring the cost of setting up additional channels. In such cases, it is desired to use as few coordination channels as possible, in contrast to exchanging as few coordination messages as possible.

The problem considered essentially corresponds to reducing the cost of setting up the test infrastructure when there are *physically distributed* ports. Typically, a distributed test architecture consists of several concurrently operating testers each of which executes a partial test sequence, which is a projection of the global test sequence that comprises only those messages which can be observed at the port assigned to the particular tester. The joint execution of the global test by sequence the testers is controlled by a Test Coordination

Procedure (TCP). In order to ensure that the global view on the SUT during testing can be inferred it is sufficient to introduce coordination messages into the partial test sequences executed by the testers. The complexity of the TCP increases with the number of testers since this introduces the cost of setting up the testers and the channels between these testers [18,19]. Once the test infrastructure is in place, the cost of sending individual messages between the testers may be very low in which case we want to minimize the cost of deploying the test infrastructure. Since the number of testers is fixed, the only opportunity for minimizing this cost is through reducing the number of channels between the testers.

This paper investigates the problem of finding a minimum number of channels such that we can overcome controllability problems by sending coordination messages through these channels. First we assume that for a message to be sent from tester t_i to tester t_j we require a channel between t_i and t_j: we do not allow this to go via another tester. We then extend this to the case where coordination messages can be sent via other testers. The problem of minimizing the number of channels has previously been considered and a greedy algorithm was proposed which adapts the test sequence generation method of [10] to distributed testing. This algorithm attempts to minimize the number of coordination channels by augmenting the data structure used in [10] and by incremental inclusion of channels required to overcome controllability problems [11].

In this paper we consider three instances of the problem. The first problem is where we have a test sequence that we wish to apply in testing and we want to use a minimum number of channels between testers in order to overcome the controllability problems in this test sequence. We then consider the problem of finding a set of channels that allows us to overcome controllability problems in any test sequence. This second problem corresponds to the situation in which we will be testing from an FSM but do not know in advance which test sequences will be used, possibly because an on-the-fly approach to test generation is being applied. Finally, we consider the problem of finding a test sequence that achieves a particular test objective and in doing so allows fewest channels to be used. We approach this third problem by defining a directed graph in which paths correspond to test sequences and the cost of a path is the size of a set of channels whose use allows controllability problems to be overcome in this sequence.

The paper differs from [11] in the following ways. The approach of [11] adapts one particular test generation algorithm [10] in an attempt to minimize the number of channels required. It initially applies the algorithm of [10] and so tries to produce a test sequence in which we do not require channels. If this fails then it adds channels in an incremental manner until the test sequence can be produced. The approach is thus a heuristic, which is based on a greedy

algorithm, that generates one particular type of test sequence. In contrast, we give *exact* methods for three general scenarios. While the first problem we consider is NP-hard, its complexity depends on the number of ports of the FSM and does not depend on the number of states it has or the size of the SUT. In almost all cases the number of ports will be small, even if there are many states, and this will allow the problem to be solved exactly. If this problem cannot be solved exactly then there is a polynomial time heuristic with an approximation ratio of $\log n$ where n is bounded above by the length of the path being considered. Note that in contrast [11] gives no information about the accuracy of the heuristic that they employ.

The paper is structured as follows. Section 2 describes multi-port FSMs and the distributed test architecture. In Section 3 we differentiate unidirectional and bidirectional channels between pairs of testers and consider the following problems: In Section 3.1 we show how we can find a minimum set of channels whose use can overcome controllability problems in a given test sequence. In Section 3.2 we then show how this can be adapted to produce a minimum set of channels whose use allows us to overcome controllability problems in any test sequence. In Section 3.3 we explore the third scenario, in which there is a test objective and we wish to find a minimum set of channels r and a test sequence $\bar{z}$ such that $\bar{z}$ achieves the test objective, it is possible to execute $\bar{z}$ without encountering controllability problems if we have channel set r, and there is no smaller set of channels which allows us to achieve the test objective without introducing controllability problems. In Section 4, we consider the same three problems in the context of transitive channels between testers. Finally, in Section 5, conclusions are drawn.

2 Preliminaries

This section describes the distributed test architecture and multi-port FSMs and gives the notation we use.

2.1 Multi-port finite state machines

Let the set $\mathcal{P} = [1, n] (= \{1, \ldots, n\})$ represent the ports of a multi-port FSM with $n > 1$ ports. A (deterministic) multi-port FSM M with $n > 1$ ports is defined by a tuple $(S, X, Y, \delta, \lambda)$ in which:

- S is the finite *set of states* of M where $s_0 \in S$ is the *initial state* of M;
- $X = \bigcup_{i=1}^{n} X_i$ is the finite *input alphabet* of M, where X_i is the input alphabet of port i, $X_i \cap X_j = \emptyset$ for all $i, j \in [1, n]$, $i \neq j$;

- $Y = \prod_{i=1}^{n}(Y_i \cup \{-\})$ is the finite *output alphabet* of M, where Y_i is the output alphabet of port i, $Y_i \cap Y_j = \emptyset$ for all $i, j \in [1, n]$, $i \neq j$, and $-$ means null output;
- δ is the *transition function* of type $S \times X \rightarrow S$; and
- λ is the *output function* of type $S \times X \rightarrow Y$.

A transition t of an FSM M is a triple $(s, s', x/y)$, where $s, s' \in S$, $x \in X$, and $y \in Y$ such that $\delta(s, x) = s'$, $\lambda(s, x) = y$. Note that $y \in Y$ is a *vector of outputs*, i.e., $y = \langle o_1, o_2, \ldots, o_n \rangle$ where $o_i \in Y_i \cup \{-\}$ for $i \in [1, n]$. s and s' are the *starting state* and the *ending state* of t, denoted $start(t)$ and $end(t)$, respectively. The *input/output pair* x/y is the *label* of t. The set of all transitions of M is denoted T.

We will use the following notation: Given a transition $t = (s, s', x/y)$,

- $port(x) = i$ if $x \in X_i$
- $inport(t) = port(x)$
- $ports(x/y) = \{port(x)\} \cup \{i | o_i \in Y_i, y = \langle o_1, o_2, \ldots, o_n \rangle, i \in [1, n]\}$
- $ports(t) = ports(x/y)$

A variable name has a bar over it (for example, $\bar{x}$) if this variable represents a sequence and ϵ denotes the empty sequence. A sequence is represented by listing its elements. For example *abc* represents the sequence with three elements: a then b and then c.

A *path* $\bar{\rho} = t_1\ t_2\ \ldots\ t_k$ $(k \geq 1)$ is a finite sequence of transitions such that if $k \geq 2$ then end(t_i) is start(t_{i+1}) for all $i \in [1, k-1]$. Path $\bar{\rho}$ is said to *start* at the state $start(t_1)$. When the ending state of the last transition of path $\bar{\rho}_1$ is the starting state of the first transition of path $\bar{\rho}_2$, we use $\bar{\rho}_1\bar{\rho}_2$ to denote the *concatenation* of paths $\bar{\rho}_1$ and $\bar{\rho}_2$. The *label* of a path $\bar{\rho} = (s_1, s_2, x_1/y_1)$ $(s_2, s_3, x_2/y_2)\ \ldots\ (s_k, s_{k+1}, x_k/y_k)$ $(k \geq 1)$ is the sequence of input/output pairs $x_1/y_1x_2/y_2\ \ldots\ x_k/y_k$, which is called an *input/output sequence*. The *input portion* and *output portion* of an input/output sequence $x_1/y_1x_2/y_2\ \ldots\ x_k/y_k$ are the input sequence $x_1x_2\ \ldots\ x_k$ and output sequence $y_1y_2\ \ldots\ y_k$, respectively. The input sequence $x_1 \ldots x_k$ (or input/output sequence $x_1/y_1x_2/y_2\ \ldots\ x_k/y_k$) is said to *label* $\bar{\rho}$. Note that we call a sequence of input/output pairs $x_1/y_1x_2/y_2\ \ldots\ x_k/y_k$, or $\bar{x}/\bar{y}$ ($\bar{x} = x_1 \ldots x_k$ and $\bar{y} = y_1 \ldots y_k$) or any combination of these an input/output sequence.

2.2 The distributed test architecture and controllability problems

In distributed testing, a *distributed test architecture* is used where a tester is placed at each port of the *system under test* (SUT) N whose externally observable behavior is modeled by a multi-port FSM M. This is illustrated in

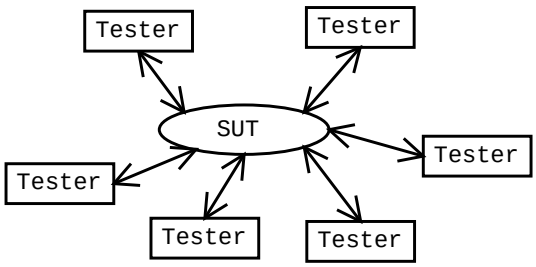

Fig. 3. A system with distributed testers

Figure 3. In this architecture, remote testers are expected to coordinate the application of a test to N. However, they cannot directly communicate with one another and there may be no global clock. These requirements can lead to controllability problems. Let us suppose that in testing N the input x at port i is expected to lead to output o at port i only and this is to be followed by the input of x' at port $j \neq i$. Then we have a controllability problem since the tester at port j does not observe either the input x or output generated by N in response to x and so does not know when to send input x' to N. In general, given an FSM M and input/output sequence $x_1/y_1x_2/y_2 \ldots x_k/y_k$ of M, a *controllability (synchronisation) problem* occurs between x_i/y_i and x_{i+1}/y_{i+1}, if $port(x_{i+1}) \notin ports(x_i/y_i)$.

Two consecutive transitions in a path $\bar{\rho}$ (or two adjacent transitions in an FSM M) t_i and t_{i+1} whose labels are x_i/y_i and x_{i+1}/y_{i+1}, form a *synchronisable pair* of transitions if t_{i+1} can follow t_i without causing a synchronisation problem. Any (sub)sequence of transitions in which every pair of consecutive transitions is synchronisable is called a *synchronisable transition (sub)sequence*. An input/output sequence is *synchronisable* if it is the label of a synchronisable transition sequence.

If there is a synchronisation problem between x_i/y_i and x_{i+1}/y_{i+1} then x_{i+1} cannot be applied after x_i when testing in the distributed test architecture since the tester at $port(x_{i+1})$ cannot know when to apply x_{i+1}. In general, the solution to the synchronisation problem is to insert a coordination message exchange relating to controllability between x_i/y_i and x_{i+1}/y_{i+1} such that the tester at a port $j \in ports(x_i/y_i)$ notifies the tester at $port(x_{i+1})$ to send its input to N [1]. However, this solution requires an (external) coordination channel from the port $j \in ports(x_i/y_i)$ to $port(x_{i+1})$. In the following sections we study the problems of minimizing the number of such channels.

3 Using bidirectional channels

It is normal to place channels between the testers and in this section we assume that two testers can only communicate if they are linked by a channel.

In Section 4 we consider the case where a tester t_1 can send a message to a tester t_2 via other testers. In this section we concentrate on the case where the channels allow communication in both directions, since most networks have this property, but indicate how the techniques can be adapted to channels in which messages can only pass in one direction.

3.1 Adding channels to overcome problems in a given test sequence

In this section we assume that we are to apply the test sequence $\bar{z} = x_1/y_1 \ldots x_k/y_k$, and we wish use a minimum number of channels so that any controllability problems can be overcome in $\bar{z}$.

The first observation is that we can identify each of the consecutive pairs $x_i/y_i x_{i+1}/y_{i+1}$ in $\bar{z}$ such that the input of x_{i+1} leads to a controllability problem and so we need to add a coordination message after x_i/y_i in $\bar{z}$. Informally, there is a controllability problem if the port at which x_{i+1} is to be sent was not involved in the previous input/output pair x_i/y_i. For each such pair we let $(P(i), p(i+1))$ denote the ordered pair such that: $P(i)$ is $ports(x_i/y_i)$ and $p(i+1)$ is $port(x_{i+1})$. That is, there is a controllability problem if we have that $p(i+1) \notin P(i)$.

Definition 1 *Given input/output sequence $\bar{z} = x_1/y_1 \ldots x_k/y_k$ we let $cp(\bar{z}) = \{(P(i), p(i+1)) | 1 \leq i < k\}$.*

In order to overcome controllability problems in $\bar{z}$, for each $(P(i), p(i+1)) \in cp(\bar{z})$ we need a channel between some port in $P(i)$ and the port $p(i+1)$. Given a pair (P, p) in which $P \subseteq \mathcal{P}$ and $p \in \mathcal{P}$ we let $links(P, p) = \{(p', p) | p' \in P\}$.

Given $(P(i), p(i+1)) \in cp(\bar{z})$ we therefore let $C_i = links(P(i), p(i+1))$ be the set of *unordered* pairs $(p, p(i+1))$ such that p is in $P(i)$: this is the set of channels such that the inclusion of any one of these is sufficient to overcome the controllability problem caused by following x_i/y_i by x_{i+1}/y_{i+1}. We can bring the notation together to give a sufficient condition for a set $C \subseteq \mathcal{P} \times \mathcal{P}$ of channels to allow us to overcome the controllability problems in $\bar{z}$, where a channel between the testers at ports p and p' is represented by the unordered pair (p, p').

Proposition 1 *Given input/output sequence $\bar{z}$, the set C of channels can be used to overcome the controllability problems in $\bar{z}$ if and only if for all $(P(i), p(i+1)) \in cp(\bar{z})$ we have that $C_i \cap C \neq \emptyset$.*

Proof
This follows from the fact that the controllability problem for some $x_i/y_i x_{i+1}/y_{i+1}$

can be overcome if and only if we have a channel from the set C_i. □

It is therefore sufficient to find the C_i and choose a set C of channels such that C contains at least one element of each C_i. This is an instance of the following problem.

Definition 2 (Hitting Set) *Given subsets $A_1, \ldots, A_j$ of a finite set Σ, $A \subseteq \Sigma$ is a hitting set if for all $1 \leq i \leq j$ we have that $A \cap A_i \neq \emptyset$. The hitting set problem is to find some smallest hitting set A for a given $A_1, \ldots, A_j$.*

The hitting set problem is known to be NP-hard [12] and so it is not surprising that our problem is also NP-hard.

Theorem 1 *Given an input/output sequence $\bar{z}$ the problem of finding a smallest set of channels that can be used to overcome controllability problems in $\bar{z}$ is NP-hard.*

Proof
We will prove this by reducing the hitting set problem to an instance of our problem. We thus assume that we want to find a smallest hitting set A for some non-empty $A_1, \ldots, A_j$ and we let $A_1 \cup \ldots \cup A_j = \{a_1, \ldots, a_n\}$. We let the port set be $\{p_1, \ldots, p_n, p_{n+1}\}$ and for all $1 \leq i \leq j$ we define two input/output pairs:

(1) x_i^1/y_i^1 where x_i^1 is input at port p_k for some $a_k \in A_i$ and y_i^1 has output at every port p_k such that $a_k \in A_i$;
(2) x_i^2/y_i^2 where x_i^2 is input at port p_{n+1} and y_i^2 has output at every port.

Now consider the input/output sequence $\bar{z} = x_1^1/y_1^1 x_1^2/y_1^2 \ldots x_j^1/y_j^1 x_j^2/y_j^2$. It is clear that no controllability problems are caused by subsequences of the form $x_i^2/y_i^2 x_{i+1}^1/y_{i+1}^1$ since y_i^2 contains output at every port. Thus, the only controllability problems occur between pairs of the form $x_i^1/y_i^1 x_i^2/y_i^2$. Further, a channel can be used to overcome the controllability problem in such a pair if and only if it connects a port p_k such that $a_k \in A_i$ to port p_{n+1}. Thus, a set A is a hitting set for $A_1, \ldots, A_j$ if and only if the set $\{(p_k, p_{n+1}) | p_k \in A\}$ of channels can be used to overcome the controllability problems in $\bar{z}$. Thus, if we can find a minimum set of channels to overcome the controllability problems in $\bar{z}$ then we have solved this instance of the hitting set problem. Finally, we can note that $\bar{z}$ and an FSM that has a path with input/output portion $\bar{z}$ can be produced in polynomial time. □

Note that the size of the problem relates to the number of ports of the SUT, rather than the number of states of an FSM specification. Thus, if we consider the number of states of the FSM we find that the complexity is of $O(1)$. Where the number of ports is not too large, it should therefore be possible to solve

this problem using exhaustive enumeration. It seems likely that in practice the number of ports will often allow this problem to be solved exactly.

While the hitting set problem is NP-hard, heuristics for this problem and its dual, the set cover problem, have been studied. In particular, it has been found that greedy algorithms are effective, having an approximation ratio of $\log n$ where n is the number of sets [5] and so is bounded above by the length of the input/output sequence $\bar{z}$.

We now outline a simple greedy algorithm that might be used if the size of the problem requires us to use approximations.

Algorithm 1 A Greedy Algorithm

Input sets $C_1, \ldots, C_j$.
Let $C = \emptyset$ and let $C_L = \{C_1, \ldots, C_j\}$
while $C_L \neq \emptyset$ **do**
  For all $(p, p') \in \bigcup_{i=1}^{j} C_i$, let $count(p, p')$ denote the number of sets from C_L that contain (p, p')
  Choose some element (p, p') that maximizes $count(p, p')$
  Set $C = C \cup \{(p, p')\}$ and remove from C_L all sets that contains (p, p')
end while
Output C

It is straightforward to adapt this technique to the case where messages can only pass in one direction through a channel. In this case, a channel is represented by an ordered pair and thus given a test sequence and a pair $(P_i, p(i+1))$ we require a channel from a port in P_i to $p(i+1)$. The problem thus remains an instance of the hitting set problem but with ordered pairs rather than unordered pairs and the greedy algorithm operates in a similar manner to the bidirectional case.

We now consider the case where we do not know the test sequence in advance and so wish to be able to use any test sequence without introducing controllability problems.

3.2 Adding channels to overcome problems in any test sequence

We have seen how, for a given test sequence, it is possible to find a smallest set of channels that will allow us to overcome all controllability problems in the test sequence. However, it may not be possible to know the test sequence to be used before implementing the test infrastructure, and where this is the case we are interested in a different problem. This problem is to find some smallest set of channels which will allow us to overcome controllability problems in any test sequence that might be used.

The approach described in this section adapts that given in Section 3.1. Essentially, we identify all pairs of adjacent transitions tt' in M. For each such pair we decide whether there is a controllability problem introduced by following t with t' and, if there is, determine the set of channels that would allow this controllability problem to be overcome. We can then phrase our problem as an instance of the hitting set problem.

Given an FSM M, we let $A(M)$ denote the set of adjacent transition pairs in M. This is defined by $A(M) = \{(t, t') \in T \times T | end(t) = start(t')\}$. We can now define the set $cp(M)$ of adjacent transition pairs of M in which there can be controllability problems and the sets of pairs of ports to be considered.

Definition 3 *Given multi-port FSM M with transition set T we have that:*

(1) $cp(M) = \{(t, t') \in A(M) | inport(t') \notin ports(t)\}$.
(2) $ReqC(M) = \{\{p\} \times P | \exists (t, t') \in cp(M) s.t. p = inport(t') \wedge P = ports(t)\}$

Proposition 2 *The set $C \subseteq \mathcal{P} \times \mathcal{P}$ of channels can be used to overcome controllability problems in any test sequence for M if and only if for all $C_i \in ReqC(M)$ we have that $C_i \cap C \neq \emptyset$.*

Proof
First assume that for all $C_i \in ReqC(M)$ we have that $C_i \cap C \neq \emptyset$. Here it is sufficient to observe that for every pair of adjacent transitions tt' of M we have that either no controllability problems are introduced by following t by t' or $ReqC(M)$ contains a set C_i of channels such that the inclusion of a channel from C_i allows t to be followed by t' without introducing controllability problems.

Now assume that the set C of channels can be used to overcome controllability problems in any test sequence for M. Then for each $C_i \in ReqC(M)$ there is some pair tt' of consecutive transitions such that it is possible to follow t by t' without causing a controllability problem if and only if C contains a channel from the set C_i and so the result follows. □

We therefore have that, for an FSM M, we can construct $ReqC(M)$ and then solve the corresponding instance of the hitting set problem. The resulting set of channels is a smallest set of channels whose use allows us to overcome controllability problems in any test sequence. Again, it is straightforward to adapt this to the use of channels that only allow messages to pass in one direction by using unordered pairs rather than ordered pairs.

In Section 3.1 we assumed that we have been given a test sequence to apply and wish to choose a minimum number of channels to overcome controllability problems in this test sequence. In Section 3.2 we considered the other extreme, in which we do not know what test sequences will be used and so require the use of a smallest set of channels that can overcome controllability problems in any test sequence that may be selected from an FSM M. In this section we consider a problem that lies between these: we have been given a test objective and wish to find an optimal test sequence and a smallest set of channels. Here, test sequence $\bar{z} = \bar{x}/\bar{y}$ and set C of channels is optimal if C can be used to overcome controllability problems in $\bar{z}$ and there is no test sequence $\bar{z}'$, that achieves the test objective, whose controllability problems can be overcome using fewer channels.

The approach will be to devise a directed graph in which we represent possible test sequences by paths. There will be multiple copies of a state s, each copy $s(t, r)$ is a vertex representing the state s having just been reached by transition t and r being the set of channels that are being used.

We only ever add an edge that represents a transition t' from vertex $s(t, r)$ if we can follow t by t'. Let us suppose that $start(t')$ is state s and $end(t')$ is state s'. There are then two cases. If we can follow t by t' without causing any controllability problems, by using only the channels that are in r, then there is an edge from vertex $s(t, r)$ to vertex $s'(t', r)$ since we do not have to add additional channels. Naturally, this is the case if and only if either $inport(t') \in ports(t)$ or there is some $(p, p') \in r$ such that $p' = inport(t')$ and $p \in ports(t)$. If we cannot follow t by t' without causing any controllability problems, by using only the channels in r, we find the set of channels whose use can overcome this controllability problem and for each such channel c there is an edge from vertex $s(t, r)$ to vertex $s'(t', r \cup \{c\})$.

We now define the directed graph $G = (V, E)$ that will form the basis for test sequence generation. First, we define the vertex set V. Note, here we give all possible vertices: it is possible that some are unreachable and so a smaller directed graph could be generated through, for example, producing it using a breadth-first search. There is thus scope for finding more efficient methods for generating G but this is not a problem we consider in this paper.

(1) For each state s, set r of channels, and transition t with ending state s, V contains a copy of s called $s(t, r)$. This represents the situation in which we have already included the cost of adding the channels in r, we are in state s, and the previous transition was t.
(2) We include the special vertex s_0, which represents being in the initial

state, not having added any channels, and having not yet applied any test input.

We can now define the set of edges for G by stating which edges are introduced for a transition t'.

(1) If t' is a transition with starting state s, label x/y and ending state s' and $s(t, r)$ is a vertex of G then we add an edge from $s(t, r)$ representing t' using the following rules:
   (a) If it is possible to follow t by t' without introducing a controllability problem, given the channel set r, then the edge has label x/y, ends in $s'(t', r)$ and has cost 0.
   (b) If it is not possible to follow t by t' without introducing a controllability problem, given the channel set r, then for every channel c from a port in $ports(t)$ to $inport(t')$ there is an edge from $s(t, r)$ to $s'(t', r \cup \{c\})$ and this edge has label x/y and cost 1. Here the cost reflects the fact that we require an additional channel.

(2) If $s = s_0$ is the starting state of t' then we include an edge from s_0 to $s'(t', \emptyset)$ that has label x/y and cost 0. This represents starting testing with the input from t'.

It is clear that if we follow a path from s_0 to a vertex $s(t, r)$, the cost is the number of channels introduced into r.

Proposition 3 *Let us suppose that $\bar{\rho}$ is a path of G that has starting vertex s_0, label $\bar{z}$, and ending vertex $s(t, r)$. Then $\bar{\rho}$ has cost $|r|$.*

Proof
This follows from the construction of G. □

We can now prove that G has the expected properties. The following two results show that the paths of G capture the possible combinations of test sequences and channel sets that cause no controllability problems.

Proposition 4 *Let us suppose that $\bar{\rho}$ is a path of G that has starting vertex s_0, label $\bar{z}$, and ending vertex $s(t, r)$. Then it is possible to apply the input portion of $\bar{z}$ in testing without causing controllability problems if we have channel set r.*

Proof
Let $\bar{z} = x_1/y_1 \ldots x_k/y_k$ and let $t_1 \ldots t_k$ denote the corresponding sequence of transitions. Thus $\bar{\rho} = (s_0, s_1(t_1, r_1)) \ldots (s_{k-1}(t_{k-1}, r_{k-1}), s_k(t_k, r_k))$ for some states $s_1, \ldots, s_k$ and $r_1, \ldots, r_k$ with $r_k = r$. Proof by contradiction: assume that it is not possible to apply $x_1 \ldots x_k$ without causing a controllability problem if we have the channels in r and let $1 \leq i < k$ be the first value such

that x_i/y_i cannot be followed by x_{i+1} without causing a controllability problem. Since $r_i \subseteq r$ we must have that r_i does not contain a channel between $port(x_{i+1})$ and a port in $ports(x_i/y_i)$. By the construction of G we therefore know that the edge from $s_i(t_i, r_i)$ to $s_{i+1}(t_{i+1}, r_{i+1})$ must be such that $r_{i+1} = r_i \cup \{c\}$ for a channel c between $port(x_{i+1})$ and a port in $ports(x_i/y_i)$. This provides a contradiction and so the result follows. □

Proposition 5 *Let us suppose that $\bar{z}$ is the label of a path $t_1 \ldots t_k$ in M that has starting state s_0, ending state s and it is possible to apply the input portion of $\bar{z} \in L(M)$ in testing without causing controllability problems if we have channel set r but not if we have channel set r' for any $r' \subset r$. Then there is a path of G that has starting vertex s_0, label $\bar{z}$, and ending vertex $s(t_k, r)$.*

Proof
We will proceed by proof by induction on the length of $\bar{z}$. Clearly the result holds for the base cases of sequences of length 0 and 1. Inductive hypothesis: assume the result holds for all sequences of length less than k, $k > 1$, and assume that $\bar{z}$ has length k. Thus, $\bar{z} = x_1/y_1 \ldots x_k/y_k$ for some $x_1, \ldots, x_k$ and $y_1, \ldots, y_k$. Let $\bar{z}' = x_1/y_1 \ldots x_{k-1}/y_{k-1}$ and so $\bar{z} = \bar{z}' x_k/y_k$.

By the inductive hypothesis, there is a path of G that has starting vertex s_0, label $\bar{z}'$, and ending vertex $s_{k-1}(t_{k-1}, r'')$ for some $r'' \subseteq r$. There are now two cases.

(1) If $port(x_k) \in ports(x_{k-1}/y_{k-1})$ or there is some channel $c = (port(x_k), p) \in r''$ such that $p \in ports(x_{k-1}/y_{k-1})$ then there is a path of G that has starting vertex s_0, label $\bar{z}$, and ending vertex $s_k(t_k, r'')$. In addition, we must have that $r = r''$ and so the result follows.
(2) Otherwise there is a channel $c = (port(x_k), p)$ such that $\{c\} = r \setminus r''$ for some $p \in ports(x_{k-1}/y_{k-1})$. By definition, G contains an edge from $s_{k-1}(t_{k-1}, r'')$ to $s_k(t_k, r'' \cup \{c\})$ as required.

□

These results tell us that for a given input/output sequence $\bar{z}$ we can determine a minimum set of channels that can be added in order to overcome controllability problems in $\bar{z}$: we find the paths of G from s_0 that have label $\bar{z}$ and choose some minimum cost path. The challenge now is to represent a test objective using G in a manner that allows us to express test generation as an optimization problem. In this section we consider one simple test objective: find a test sequence that executes a given transition t.

In G, the nodes of the form $s(t, r)$ represent situations in which t has just been executed. Let us suppose that $\bar{\rho}$ is a minimum cost path in G from s_0 to a vertex of the form $s(t, r)$, let $s(t, r_1)$ denote the ending vertex of $\bar{\rho}$, and

let $\bar{z}$ be the label of $\bar{\rho}$. Then r_1 denotes a minimum size set of channels that allows us to execute $\bar{z}$ without introducing controllability problems. We thus require a minimum cost path in G from s_0 to some $s(t, r)$. We can represent the problem of finding such a path as a directed graph optimization problem.

Given G and transition t let G_t denote the directed graph formed from G by adding a new vertex v_t and an edge from each vertex of the form $s(t, r)$ to v_t, such edges being given cost 0.

Proposition 6 *An input/output sequence $\bar{z}$ is a sequence that allows transition t to be executed with the addition of a minimum number of channels, in order to overcome controllability problems, if and only if $\bar{z}$ is the label of a minimum cost path from s_0 to v_t in G_t.*

Proof
Note that a path in G from s_0 to some $s(t, r)$ has cost $|r|$. In addition, it is clearly sufficient to only consider paths of M that end in t.

First assume that $\bar{z}$ is a sequence that allows t to be executed with the addition of a minimum number of channels, in order to overcome controllability problems. Let r denote some such minimum set of channels. By Proposition 5, there is a path in G from s_0 to $s(t, r)$ with label $\bar{z}$ and so there is a path from s_0 to v_t with cost $|r|$ and label $\bar{z}$. By Proposition 4, if $\bar{\rho}$ is a path of G that has starting vertex s_0, label $\bar{z}$, and ending vertex $s(t, r)$ then it is possible to apply the input portion of $\bar{z}$ in testing without causing controllability problems if we have channel set r. Thus, by the minimality of $\bar{z}$ there is no path from s_0 in G to a vertex $s(t, r')$ for a set r' of channels such that $|r'| < |r|$ and thus $\bar{z}$ is the label of a minimum cost path from s_0 to v_t as required.

Now assume that $\bar{z}$ is the label of a minimum cost path from s_0 to v_t and we require to prove that $\bar{z}$ is a sequence that allows transition t to be executed with the addition of a minimum number of channels, in order to overcome controllability problems. Let r denote some minimum set of channels that can be used with $\bar{z}$ and thus one minimum cost path from s_0 to v_t passes through vertex $s(t, r)$. Proof by contradiction: assume that $\bar{z}'$ is a sequence that allows transition t to be executed with the addition of a set r' of channels with $|r'| < |r|$. By Proposition 5, there is a path in G from s_0 to $s(t, r')$ and thus to v_t and clearly this has a lower cost than the path with label $\bar{z}$ that reaches v_t by passing through $s(t, r)$, providing a contradiction as required. □

Test generation for other criteria is left for future work. However, it is relatively straightforward to adapt G in order to represent the problem of finding a path that includes a given path $\bar{\rho}$ using fewest channels. If the start state of $\bar{\rho}$ is s_0 then the problem reverts to that described in Section 3.1. If $s \neq s_0$ is the start state of $\bar{\rho}$ then for each vertex $s(t, r)$ we can find the smallest number

$c(r, t, \bar{\rho})$ of channels we can add to r in order for us to be able to follow t with $\bar{\rho}$. The value of $c(r, t, \bar{\rho})$ can be computed by adapting the approach given in Section 3.1. We can therefore add a new vertex $v_{\bar{\rho}}$ and an edge from $s(t, r)$ to $v_{\bar{\rho}}$ with cost $c(r, t, \bar{\rho})$. Then, a minimum cost path from s_0 to $v_{\bar{\rho}}$ represents an optimal test sequence and set of channels for the given path $\bar{\rho}$. Importantly, many test criteria can be represented in terms of including certain paths in a test sequence.

Now consider the case where a channel allows messages to pass in one direction only and so channels are represented by ordered pairs. Then a vertex in the directed graph is in the form $s(t, r)$ where r is a set of ordered pairs rather than unordered pairs. An edge from $s(t, r)$ representing a transition t' goes to a vertex $s'(t', r)$ if either tt' is synchronisable or there is a channel in r from a port in $ports(t)$ to $inport(t')$. Otherwise, there is an edge from $s(t, r)$ representing a transition t' to each vertex $s'(t', r')$ such that $r' = r \cup \{(p, p')\}$ for $p' = inport(t')$ and a port p in $ports(t)$. The optimization problem can then be handled in an identical manner.

We now consider the case in which a tester can send a message to another tester via other testers.

4 Transitive channels

In this section we assume that the channels are bidirectional but that a tester t can send a message to a tester t' via other testers. For example, t could send a message to a tester t_1, who sends a message to a tester t_2 who then sends the message to t'. Naturally, the methods given in this section can easily be extended to channels that allow messages in one direction only and can be adapted in a similar manner to the case already considered.

Let us suppose that we have a set C of channels between the testers, and so each channel is represented by an unordered pair. Then C defines a relation between ports: ports p and p' are related if and only if $(p, p') \in C$. Let $\sim_C$ denote the transitive reflexive closure of this relation, and thus ports p and p' are related under $\sim_C$ if either $p = p'$ or there exists $(p_1, p_2), \ldots, (p_l, p_{l+1})$ such that $p = p_1$, $p' = p_{l+1}$ and for all $1 \leq i \leq l$ we have that $(p_i, p_{i+1}) \in C$. The following is clear.

Proposition 7 *The testers at ports p and p' can communicate using the set C of channels if and only $p \sim_C p'$.*

It is clear that $\sim_C$ is an equivalence relation and so it defines a partition $\wp_C$ of the set $\mathcal{P}$ of ports: two ports p and p' are in the same set of $\wp_C$ if and only

if $p \sim_C p'$.

In the following, we adapt the solutions for the three problems considered in Section 3 to the case where a tester can send a message to another tester via other testers. In the first problem, we wish to apply the test sequence $\bar{z} = x_1/y_1 \ldots x_k/y_k$ using a minimum number of channels to overcome the controllability problems in $\bar{z}$. For this we want a smallest set C of channels such that for each $(P(i), p(i+1)) \in cp(\bar{z})$ there is a port $p \in P(i)$ such that $p \sim_C p(i+1)$. Hence, it is necessary to connect some pair in $C_i = links(P(i), p(i+1))$ and thus we need some set C of channels such that there exists $(p, p') \in C_i$ such that $p \sim_C p'$. We can now adapt the proposition and the algorithm given in Section 3.1 as follows:

- replace "we have that $C_i \cap C \neq \emptyset$" by "we have that there is some $(p, p') \in C_i$ such that $p \sim_C p'$" in Proposition 1.
- For Algorithm 1, in step 4 define $count(p, p')$ as the number of sets from C_L that contain a pair (p_1, p_2) such that $p_1 \sim_{C \cup \{(p,p')\}} p_2$ and in step 6 replace "contains (p, p')" by "contains a pair (p_1, p_2) such that $p_1 \sim_C p_2$".

We can extend this to the case where we want a smallest set of channels that allows any test sequence to be applied without encountering controllability problems. As in Section 3.2, we base this on the set $cp(M)$ of pairs of adjacent transitions for which there is a controllability problem. Recall that $ReqC(M) = \{\{p\} \times P | \exists (t, t') \in cp(M) s.t. p = inport(t') \wedge P = ports(t)\}$.

Proposition 8 *The set $C \subseteq \mathcal{P} \times \mathcal{P}$ of channels can be used to overcome controllability problems in any test sequence for M when testers can communicate via other testers if and only if for all $C_i \in ReqC(M)$ we have some $(p, p') \in C_i$ such that $p \sim_C p'$.*

We therefore have that, for an FSM M, we can construct $ReqC(M)$ and then solve the corresponding optimisation problem. The resulting set of channels is a smallest set of channels whose use allows us to overcome controllability problems in any test sequence. It is straightforward to adapt the greedy algorithm described above.

4.1 Finding a test sequence that requires fewest channels

In this section we consider the following problem: we have been given a test objective and wish to find an optimal test sequence and a smallest set of channels where testers can communicate via other testers. We adapt the approach given in Section 3.3. Again there will be multiple copies of a state s, each copy $s(t, \wp)$ is a vertex representing the state s having just been reached by

transition t and $\wp$ being a partition of the set of ports. We only ever add an edge that represents a transition t' from vertex $s(t, \wp)$ if we can follow t by t'. As we have seen, a set C of channels defines a partition $\wp_C$ of the set of ports. In addition, given a partition $\wp$ of the set of ports it is possible to find a smallest set of channels C such that $\wp_C = \wp$.

Proposition 9 *Given a partition $\wp$ of $\mathcal{P}$, a smallest set C of channels such that $\wp_C = \wp$ has $n - |\wp|$ channels.*

The following algorithm takes as input a partition $\wp$ and outputs a set C of channels such that $\wp = \wp_C$.

Algorithm 2 Producing channels for a partition

Input set $\mathcal{P}$ of ports and partition $\wp$
Set $C = \emptyset$
while $\wp_C \neq \wp$ **do**
  Find some pair (p, p') of ports such that p and p' are in the same set in $\wp$ but in different sets of $\wp_C$.
  Add (p, p') to C
end while
Output C

If $C = \emptyset$ then we have the partition, denoted $\wp_0 = \{\{p\} | p \in \mathcal{P}\}$ in which each set is a singleton.

Let us suppose that $start(t')$ is state s and $end(t')$ is state s' and we have vertex $s(t, \wp)$. There are two cases.

(1) We can follow t by t' without causing any controllability problems, by using any set C of channels such that $\wp_C = \wp$. Then there is an edge from vertex $s(t, \wp)$ to vertex $s'(t', \wp)$ since we do not have to add additional channels. Naturally, this is the case if and only if either $inport(t') \in ports(t)$ or there is some $\pi \in \wp$ and $p \in ports(t)$ such that p and $inport(t')$ are both in π.
(2) We cannot follow t by t' without causing any controllability problems, by using any set C of channels such that $\wp_C = \wp$. Then there is an edge from vertex $s(t, \wp)$ to vertex $s'(t', \wp')$ for all $\wp'$ that can be formed from $\wp$ by merging one set $\pi_1 \in \wp$ such that $inport(t') \in \pi_1$ and another set $\pi_2 \in \wp$ such that $\pi_2 \cap ports(t) \neq \emptyset$.

We now define the directed graph $G^T = (V^T, E^T)$ that will form the basis for test sequence generation. First, we define the vertex set V^T. Again, we give all possible vertices: it is possible that some are unreachable.

(1) For each state s, partition $\wp$ of the set of channels, and transition t with ending state s, V^T contains a copy of s called $s(t, \wp)$. This represents the

situation in which we have already included the cost of adding a minimum set of channels that define $\wp$, we are in state s, and the previous transition was t.

(2) We include the special vertex s_0, which represents being in the initial state, not having added any channels, and having not yet applied any test input.

We can now define the set E^T of edges by stating which edges are introduced for a transition t'.

(1) If t' is a transition with starting state s, label x/y and ending state s' and $s(t, \wp)$ is a vertex in V^T then we add edges from $s(t, \wp)$ representing t' using the following rules:
   (a) If it is possible to follow t by t' without introducing a controllability problem, given channels that define $\wp$, then the edge with label x/y ends in $s'(t', \wp)$. This edge has cost 0.
   (b) Otherwise for every channel $c = (p, p')$ from a port in $ports(t)$ to $inport(t')$ there is an edge from $s(t, \wp)$ to $s'(t', \wp')$ for $\wp'$ formed from $\wp$ by merging set $\pi_1 \in \wp$ such that $p \in \pi_1$ and set $\pi_2 \in \wp$ such that $p' \in \pi_2$. This edge has label x/y and cost 1.

(2) If $s = s_0$ is the starting state of t' then we include an edge from s_0 to $s'(t', \wp_0)$ that has label x/y and cost 0. This represents starting testing with the input from t'.

It is clear that if we follow a path from s_0 to a vertex $s(r, \wp)$, the cost is the minimum number of channels that can be used to define $\wp$.

Proposition 10 *Let us suppose that $\bar{\rho}$ is a path of G^T that has starting vertex s_0, label $\bar{z}$, and ending vertex $s(t, \wp)$. Then $\bar{\rho}$ has cost $n - |\wp|$.*

Proof
This follows from the construction of G^T. □

We now prove that G^T has the expected properties. The proofs of the following two results are similar to those of Propositions 4 and 5 respectively.

Proposition 11 *Let us suppose that $\bar{\rho}$ is a path of G^T that has starting vertex s_0, label $\bar{z}$, and ending vertex $s(t, \wp)$. Then it is possible to apply the input portion of $\bar{z}$ in testing without causing controllability problems if we have a channel set C such that $\wp_C = \wp$ and testers can communicate via other testers.*

Proposition 12 *Let us suppose that $\bar{z}$ is the label of a path $t_1 \ldots t_k$ in M that has starting state s_0, ending state s and it is possible to apply the input portion of $\bar{z} \in L(M)$ in testing without causing controllability problems if we have channel set C but not if we have channel set C' with $|C'| < |C|$. Then*

there is a path of G^T that has starting vertex s_0, label $\bar{z}$, and ending vertex $s(t_k, \wp_C)$.

These results tell us that for a given input/output sequence $\bar{z}$ we can find an optimal partition and we can then use Algorithm 2 to find a minimum set of channels: we find the paths of G^T from s_0 that have label $\bar{z}$ and choose a minimum cost path. We can therefore represent test generation in terms of an optimization problem and consider the problem of executing a given transition. Given G^T and transition t we let G^T_t denote the directed graph formed from G^T by adding a new vertex v_t and an edge from each vertex of the form $s(t, \wp)$ to v_t, such edges being given cost 0. The proof of the following is similar to the proof of Proposition 6.

Proposition 13 *An input/output sequence $\bar{z}$ is a sequence that allows transition t to be executed with the addition of a minimum number of channels when testers can communicate via other testers if and only if $\bar{z}$ is the label of a minimum cost path from s_0 to v_t in G^T_t.*

5 Conclusions

When testing a system that has physically distributed interfaces/ports we place a tester at each port. Each tester observes only the events at its own port and as a consequence there can be additional controllability problems. It is known that these controllability problems can be overcome if we introduce a network through which the testers can exchange external coordination messages but the introduction of such a network can increase the cost of testing. This paper has therefore considered the problem of adding a minimum number of channels between testers in order to overcome controllability problems.

We have considered three alternative scenarios and corresponding problems. In the first scenario we have a test sequence that we wish to apply and want to use a minimum number of channels between the testers in order to overcome controllability problems in this test sequence. We have shown that this problem is NP-hard and can be seen as an instance of the hitting set problem. While this problem is NP-hard, the size of the problem depends on the number of ports of the SUT, not on the number of states of the model that represents its required behaviour: it seems likely that in most cases the number of ports is sufficiently small to produce optimal solutions. In addition, greedy algorithms are known to be effective for the hitting set problem and have an approximation ratio of $\log n$, where n is bounded above by the length of the sequence being considered. The second scenario is that we wish to introduce an external network but do not yet know what test sequences or test objectives will be used. In this case we want to find a minimum number of channels that allow controllabilty problems

to be overcome in any test sequence and it transpires that this can also be seen as an instance of the hitting set problem.

The third scenario is that we know the test objective but can choose any test sequence that achieves this. We have shown how a directed graph G can be devised, based on a finite state machine model of the SUT, in which we can represent the problem of finding an optimal test sequence and set of channels. Specifically, each path through G represents a possible test sequence and a set of channels whose use allows us to overcome all controllabilty problems in that test sequence. In addition, the cost of a path in G is the number of channels included in that path. We have investigated one test objective, that we wish to execute a given transition, and have shown how we can represent the problem of finding a corresponding test sequence and minimal set of channels in terms of finding a minimum cost path in an augmented version of G. It seems likely that many other test criteria can be represented in a similar way.

We have considered three types of networks. In the first two we have that two testers can exchange external coordination messages through the network if and only if there is a channel between them. These two cases differ in the nature of the channel: whether it allows messages in one direction only or in two directions. In the third type of network two testers can communicate via other testers.

There remain several possible topics for future work. First, for the situation in which we have a test criterion, we could consider alternative criteria. Second, we have only considered testing from deterministic finite state machines and it would be interesting to investigate alternative formalisms and, in particular, nondeterminism. Finally, recent work has defined models in which a transition can be triggered by the SUT receiving input at several ports [8] and similar problems should exist for such models.

Acknowledgements

This work was supported in part by Natural Sciences and Engineering Research Council (NSERC) of Canada grant number OGP00000976, Testing State Based Systems, and the EU Framework 6 grant: Training and Research on Testing (TAROT) — a Marie-Curie Research Training Network.

References

[1] L. Cacciari and O. Rafiq. Controllability and observability in distributed testing. *Information and Software Technology*, 41(11–12):767–780, 1999.

[2] J. Chen, R. M. Hierons, and H. Ural. Conditions for resolving observability problems in distributed testing. In *24rd IFIP International Conference on Formal Techniques for Networked and Distributed Systems (FORTE 2004)*, volume 3235 of *Lecture Notes in Computer Science*, pages 229–242. Springer-Verlag, 2004.

[3] J. Chen, R. M. Hierons, and H. Ural. Resolving observability problems in distributed test architecture. In *25th IFIP International Conference on Formal Techniques for Networked and Distributed Systems (FORTE 2005)*, volume 3731 of *Lecture Notes in Computer Science*, pages 219–232. Springer-Verlag, 2005.

[4] W.-H. Chen and H. Ural. Synchronizable test sequences based on multiple UIO sequence. *IEEE/ACM Transactions on Networking*, 3(2):152–157, 1995.

[5] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[6] R. Dssouli and G. von Bochmann. Error detection with multiple observers. In *Protocol Specification, Testing and Verification V*, pages 483–494. Elsevier Science (North Holland), 1985.

[7] R. Dssouli and G. von Bochmann. Conformance testing with multiple observers. In *Protocol Specification, Testing and Verification VI*, pages 217–229. Elsevier Science (North Holland), 1986.

[8] S. Haar, C. Jard, and G.-V. Jourdan. Testing input/output partial order automata. In *19th IFIP TC6/WG6.1 International Conference on Testing of Software and Communicating Systems (TestCom/FATES 2007)*, volume 4581 of *Lecture Notes in Computer Science*, pages 171–185, Tallinn, Estonia, 26–29 June 2007. Springer.

[9] 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.

[10] R. M. Hierons and H. Ural. Reduced length checking sequences. *IEEE Transactions on Computers*, 51(9):1111–1117, 2002.

[11] G. V. Jourdan, H. Ural, and H. Yenigun. Minimizing coordination channels in distributed testing. In *26th IFIP International Conference on Formal Techniques for Networked and Distributed Systems (FORTE 2006)*, volume 4229 of *Lecture Notes in Computer Science*, pages 451–466. Springer-Verlag, 2006.

[12] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*. Plenum Press, New York-London, 1972. 85–103.

[13] A. Khoumsi. A temporal approach for testing distributed systems. *IEEE Transactions on Software Engineering*, 28(11):1085–1103, 2002.

[14] 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.

[15] 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.

[16] O. Rafiq and L. Cacciari. Coordination algorithm for distributed testing. *The Journal of Supercomputing*, 24(2):203–211, 2003.

[17] K.-C. Tai and Y.-C. Young. Synchronizable test sequences of finite state machines. *Computer Networks and ISDN Systems*, 30(12):1111–1134, 1998.

[18] Andreas Ulrich and Hartmut König. Architectures for testing distributed systems. In *12th IFIP International Workshop on Testing of Communicating Systems (IWTCS 99)*, pages 93–108, Budapest, Hungary, September 1999.

[19] T. Walter, I. Schieferdecker, and J. Grabowski. Test architectures for distributed systems: state of the art and beyond. In *11th IFIP International Workshop on Testing of Communicating Systems (IWTCS 98)*, pages 149–174, Tomsk, Russia, 1998.