Université d'Ottawa · University of Ottawa

PERMISSION DE REPRODUIRE ET DE DISTRIBUER LA THÈSE

PERMISSION TO REPRODUCE AND DISTRIBUTE THE THESIS

NOM DE L'AUTEUR / NAME OF AUTHOR:	WILLIAMS, Craig
ADRESSE POSTALE / MAILING ADDRESS:	3275 RIVERSIDE DRIVE OTTAWA ON K1V8N9
GRADE / DEGREE:	**ANNÉE D'OBTENTION / YEAR GRANTED**
MCS (Computer Science)	2003
TITRE DE LA THÈSE / TITLE OF THESIS: CHECKING SEQUENCES FOR DISTRIBUTED TEST ARCHITECTURES	

L'auteur permet, par la présente, la consultation et le prêt de cette thèse en conformité avec les règlements établis par le bibliothécaire en chef de l'Université d'Ottawa. L'auteur autorise aussi l'Université d'Ottawa, ses successeurs et cessionnaires, à reproduire cet exemplaire par photographie ou photocopie pour fins de prêt ou de vente au prix coûtant aux bibliothèques ou aux chercheurs qui en feront la demande.

Les droits de publication par tout autre moyen et pour vente au public demeureront la propriété de l'auteur de la thèse sous réserve des règlements de l'Université d'Ottawa en matière de publication de thèses.

The author hereby permits the consultation and the lending of this thesis pursuant to the regulations established by the Chief Librarian of the University of Ottawa. The author also authorizes the University of Ottawa, its successors and assignees, to make reproductions of this copy by photographic means or by photocopying and to lend or sell such reproductions at cost to libraries and to scholars requesting them.

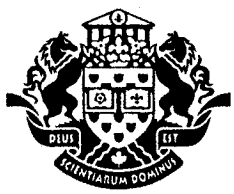
The right to publish the thesis by other means and to sell it to the public is reserved to the author, subject to the regulations of the University of Ottawa governing the publication of theses.

N.B. LE MASCULIN COMPREND ÉGALEMENT LE FÉMININ

3 March 2003

DATE

(AUTEUR) SIGNATURE (AUTHOR)



Université d'Ottawa · University of Ottawa

Université d'Ottawa · University of Ottawa

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

WILLIAMS, Craig P.L.

AUTEUR DE LA THÈSE - AUTHOR OF THESIS

M.C.S.

GRADE - DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT - FACULTY, SCHOOL, DEPARTMENT

TITRE DE LA THÈSE - TITLE OF THE THESIS

Checking Sequences for Distributed Test Architectures

Hasan Ural

DIRECTEUR DE LA THÈSE - THESIS SUPERVISOR

EXAMINATEURS DE LA THÈSE - THESIS EXAMINERS

M. Barbeau

G. von Bachmann

J.-M. De Koninck, Ph.D.

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

SIGNATURE

DEAN OF THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Checking Sequences for Distributed Test Architectures

Craig P.L. Williams

A Thesis
*Submitted to the Faculty of Graduate and PostDoctoral Studies of the University of Ottawa in Partial Fulfillment of the Requirements for the Degree of Masters in Computer Science.**

School of Information Technology and Engineering
University of Ottawa
Ottawa, Ontario

---

* The Masters program in Computer Science is a joint program with Carleton University, administered by the Ottawa-Carleton Institute for Computer Science

Abstract

The objective of testing is to determine whether an implementation under test conforms to its specification. In distributed test architectures involving multiple testers, this objective can be complicated by the fact that testers may encounter problems relating to controllability and observability during the application of a checking sequence. A controllability problem, also known as a synchronization problem, exists when a tester cannot determine when to send a particular input to the IUT. An observability problem exists when a tester cannot determine whether a particular output has been received from the IUT in response to the related input in the specification. Solutions in the literature to these problems generally require either appending additional input-output sequences or the use of external coordination messages between testers.

This thesis proposes two methods for constructing checking sequences with no potential controllability and observability problems. The first method assumes the presence of a reliable reset in the implementation and constructs digraphs to facilitate the construction of state and transition cover subsequences. A Rural Chinese Postman Path on the final digraph constructed by the method yields a synchronizable ordering of these subsequences. The second method does not require a reliable reset and constructs digraphs to generate subsequences that verify each state and transition. An Euler Tour of the final digraph constructed by the method yields a checking sequence. In both methods, by considering controllability and observability problems during the construction of the checking sequence, the use of coordination messages is either minimized or, if possible, avoided altogether.

Acknowledgements

I would like to acknowledge my supervisor, Dr. Hasan Ural, for introducing me to the area of protocol conformance testing. My thanks to him for his guidance, patience and support.

My studies would not have been possible without the financial support of the Natural Sciences and Engineering Research Council of Canada and the University of Ottawa.

Table of Contents

List of Figures

List of Tables

Chapter One

Introduction

1.1 Background

The objective of protocol conformance testing is to determine, by means of testing, if a protocol implementation conforms to its specification. Testing is carried out in a test architecture that consists of testers and the implementation under test (IUT). Both the OSI [II95] and ODP [IS95] models define a number of abstract test architectures.

The majority of testing literature assumes a test architecture consisting of one tester that applies inputs and observes outputs at a single point of control and observation, or "port." Distributed test architectures, which consist of multiple ports and multiple testers, introduce the possibility of coordination problems among the testers during testing, referred to in the literature as the problems of controllability and observability. A controllability problem, also known as a synchronization problem, exists when a tester cannot determine when to send a particular input to the IUT. The literature proposes several methods for constructing synchronizable input-output sequences, i.e. a test sequence or checking sequence without any synchronization problems. An observability problem exists when a tester cannot determine whether a particular output has been received from the IUT in response to the related input in the specification. The literature proposes several methods for resolving observability problems, which generally require either direct communication among the testers, or additional input-output sequences. In this thesis we propose two methods for generating checking sequences which can be

applied in a distributed test architecture without encountering problems relating to controllability or observability.

1.2 Motivation and Objectives of the Thesis

The problem that will be studied, in its most general form, is how to detect and eliminate potential controllability and observability problems that may be encountered while testing an implementation against a Finite State Machine (FSM) representation of a communication protocol specification in an ISO/ODP distributed test architecture. Many researchers have studied the problems of controllability and observability, and a review of these studies is included later in this thesis. Earlier studies on the observability problem propose solutions where first a synchronizable test sequence or checking sequence is generated, and then inspected for observability problems. These problems are then resolved by augmenting the sequence with additional sequences or direct communication between testers, depending on the test architecture. More recent solutions consider potential controllability and observability problems during the construction of a test sequence, thereby minimizing the need for corrective action. However, such methods are currently limited to a fault model consisting only of output faults.

Our review of these solutions has led us to study a method that identifies potential controllability and observability problems prior to constructing a checking sequence, in an effort to minimize the amount of corrective action needed for the elimination of these problems. Thus, the objective of this thesis is to propose methods for generating a synchronizable checking sequence whereby the number of corrective actions needed to

eliminate controllability and observability problems are minimized, and that can be applied in a distributed test architecture without encountering these problems.

1.3 Contributions of the Thesis

We propose two methods that generate checking sequences with no possibility of potential controllability or observability problems when applied to a 2-port FSM in a distributed test architecture. These methods can be used to generate checking sequences that can be applied in distributed test architectures where testers are able to communicate amongst themselves directly or via external coordination messages.

The first method assumes the presence of a reliable reset in the implementation and constructs digraphs to facilitate the construction of state and transition cover subsequences. A Rural Chinese Postman Path on the final digraph constructed by the method yields a synchronizable ordering of these subsequences. The second method does not require a reliable reset and constructs digraphs to generate subsequences that verify each state and transition. An Euler Tour of the final digraph constructed by the method yields a checking sequence. In both methods, by considering controllability and observability problems during the construction of the checking sequence, the use of coordination messages is either minimized or, if possible, avoided altogether.

We prove that both proposed methods ensure the absence of potential controllability and observability problems.

Finally, we extend the proposed methods so that they can be applied to an n-port FSM in a distributed test architecture, $n > 2$.

1.4 Organization of the Thesis

Chapter 2 outlines the preliminaries needed to describe the proposed methods, including the FSM model, graph-theoretic definitions, and an introduction to protocol conformance testing. Chapter 3 reviews the existing methods in literature that have been proposed for generating checking sequences, and existing methods that have been proposed to solve controllability and observability problems. Chapter 4 describes our proposed methods and compares them with the existing methods. Chapter 5 presents our conclusions, with a summary of contributions and directions for future research.

Chapter Two

Preliminaries

2.1 FSM Model and its Graphical Representation

An ***n*-port Finite State Machine (*np*-FSM) M is a sextuple (S, Σ, Γ, δ, λ, s_1)** where

- S is a finite set of states of M,
- $s_1 \in S$ is the initial state of M,
- Σ is an n-tuple $(\Sigma_1, \Sigma_2, \ldots, \Sigma_n)$, where Σ_k is the input alphabet of port k, and $\Sigma_i \cap \Sigma_j = \varnothing$ for $i \neq j$, $i, j = 1, 2, \ldots, n$. Let $I = \Sigma_1 \cup \Sigma_2 \cup \ldots \cup \Sigma_n \cup \{-\}$, where - means *null* input,
- Γ is an n-tuple $(\Gamma_1, \Gamma_2, \ldots, \Gamma_n)$, where Γ_k is the output alphabet of port k, and $\Gamma_i \cap \Gamma_j = \varnothing$ for $i \neq j$, $i, j = 1, 2, \ldots, n$. Let $O = \{<a_1, a_2, \ldots, a_n> \mid \exists\, a_i \in \Gamma_i \cup \{-\}, 1 \leq i \leq n\}$, where - means *null* output,
- δ is the transition function that maps $S \times I$ to S, i.e., δ: $S \times I \rightarrow S$, and
- λ is the output function that maps $S \times I$ to O, i.e., λ: $S \times I \rightarrow O$.

An FSM M is **deterministic** if, for each input $x \in I$, there is at most one transition defined at each state of M. Functions δ and λ are extended from single input symbols to input sequences: if s_1 is a state and $X = i_1 \ldots i_k$ is an input sequence of M, $X \in I^*$, then $\delta(s_1, X) = \delta(\delta(s_1, i_1), i_2 \ldots i_k) = s_{k+1}$ (with $s_{j+1} = \delta(s_j, i_j), j = 1, \ldots, k$), and $\lambda(s_1, X) = \lambda(s_1, i_1)$ $\lambda(s_2, i_2 \ldots i_k) = o_1 \ldots o_k$ (with $o_j = \lambda(s_j, i_j), j = 1, \ldots, k$).

An FSM M is said to be **minimal** if none of its states are equivalent (i.e., $\forall\, s_i, s_j \in S$, $s_i \neq s_j$, $\exists$ an input sequence $X \in I^*$ such that $\lambda(s_i, X) \neq \lambda(s_j, X)$). An FSM M is said to be **completely specified** if, for each input $x \in I$, there is a transition defined at each state of M.

An FSM M can be represented by a directed graph $G = (V, E)$ where a set of vertices V represents the set S of states of M, and a set of directed edges E represents all specified transitions of M. A **transition** of an *np*-FSM M is a triple $t_{jk} = (s_j, s_k; x/y)$, where $s_j, s_k \in S$, $x \in I$, and $y \in O$ such that $\delta(s_j, x) = s_k$, $\lambda(s_j, x) = y$, and x/y is known as an **input/output pair**. Each edge $e_{jk} = (v_j, v_k; x/y) \in E$ represents a state transition from state s_j to state s_k with input x and output y where the input/output pair x/y is the **label** of e_{jk}, denoted by $label(e_{jk})$, v_j is called the **head** of e_{jk}, denoted by $head(e_{jk})$, and v_k is called the **tail** of e_{jk}, denoted by $tail(e_{jk})$.

A **path** $P = (v_1, v_2; x_1/y_1)(v_2, v_3; x_2/y_2)\ldots(v_{k-1}, v_k; x_{k-1}/y_{k-1})$, $k > 1$, in $G = (V, E)$ is a finite sequence of adjacent (but not necessarily distinct) edges in G, where v_1 and v_k are $head(P)$ and $tail(P)$, and $x_1/y_1, x_2/y_2, \ldots, x_{k-1}/y_{k-1}$ is the **label** of P, denoted $label(P)$. A path P is represented by $(v_1, v_k; X/Y)$ where $label(P) = X/Y$ is the **input-output sequence** $(x_1/y_1)(x_2/y_2)\ldots(x_{k-1}/y_{k-1})$, input sequence $X = (x_1x_2\ldots x_{k-1})$ is the **input portion** of X/Y, and output sequence $Y = (y_1y_2\ldots y_{k-1})$ is the **output portion** of X/Y. The **cost** or **length** of each edge of G is equal to the number of input/output pairs in its label. The cost of a path (or length of a path) P in G is the sum of the costs (or lengths) of edges included in P and is denoted $cost(P)$. The first transition $(v_1, v_2; x_1/y_1)$ of path P is denoted $first(P)$ and the last transition $last(P)$. The **concatenation** of a path A and a path B is denoted A@B.

A sequence $(i_1\, i_2 \ldots i_k)$ is a **subsequence** of $(x_1\, x_2 \ldots x_m)$ if there exists a Δ, $0 \le \Delta \le m-k$, such that for all j, $1 \le j \le k$, $i_j = x_{j+\Delta}$. A sequence $(i_1\, i_2 \ldots i_k)$ is a **prefix** of $(x_1\, x_2 \ldots x_m)$ if $\forall\, j$, $1 \le j \le k$, $i_j = x_j$.

An FSM M has a **reset** function if there exists an input $r \in I$ which takes M from any state s_i to the initial state s_1 with a single transition $(s_i, s_1; r/-)$.

A **synchronizing sequence** of an FSM M is an input sequence that takes M to a specified final state, regardless of the output or the initial state. A synchronizing sequence may not exist for every minimal FSM [KZ78].

A digraph $G = (V, E)$ is **strongly connected** if, for any pair of vertices v_j and v_k, there exists a path from v_j to v_k. It is **weakly connected** if its underlying undirected graph is connected. A **tour** of G is a path in G that starts and ends at the same vertex of G. An **Euler tour** of G is a tour that contains every edge of E exactly once. A **postman tour (PT)** of G is a tour that contains every edge in E at least once. A **rural postman tour (RPT)** of G over a set $E_C \subseteq E$ is a tour traversing every edge in E_C at least once. A **Chinese postman tour (CPT)** is a minimum-cost tour that contains every edge in E at least once. A **rural Chinese postman tour (RCPT)** of G over a set $E_C \subseteq E$ is a minimum-cost tour that traverses every edge in E_C at least once. A **rural postman path (RPP)** from v_i to v_j over $E_C \subseteq E$ is a path from v_i to v_j that includes every edge in E_C. A **rural Chinese postman path (RCPP)** from v_i to v_j over $E_C \subseteq E$ is a minimum-cost RPP. The **edge-induced subgraph** of $G[E_C]$ is the subgraph of G whose vertex set is the set of ends of edges in E_C and whose edge set is E_C. $G[E_C]$ is an edge-induced **spanning** subgraph of G if its vertex set is V.

Given a vertex $v \in V$, the **in-degree** of v, $d_i(v)$, is defined as $|\{(u, v): (u, v) \in E\}|$ and the **out-degree** of v, $d_o(v)$, is defined as $|\{(v, w): (v, w) \in E\}|$. A vertex v is a **source** if $d_i(v) = 0$, and v is a **sink** if $d_o(v) = 0$. A digraph $G = (V, E)$ is **symmetric** if $d_i(v) = d_o(v)$, $\forall\, v \in V$. Given a postman tour P of G, let $\chi(v_j, v_k; x/y) \geq 1$ be the number of times edge $(v_j, v_k; x/y)$ is contained in P. If edge $(v_j, v_k; x/y)$ is replicated $\chi(v_j, v_k; x/y)$ times, a symmetric graph $G^{\wedge}$, called a **symmetric augmentation** of G, is obtained and P is an

Euler tour of $G^{\wedge}$. A **minimal symmetric augmentation** of G is achieved by replicating edges of G such that $G^{\wedge}$ is symmetric and the number of replicated edges are minimized.

2.2 Protocol Conformance Testing

The aim of protocol conformance testing is to determine, by means of testing, if a protocol implementation conforms to its specification [IS94]. A **test sequence** is an input-output sequence, starting at the initial state, which checks whether each transition of the FSM is correctly implemented in a given implementation. Typically, protocol conformance testing involves constructing a test sequence from the protocol specification, applying the test sequence to the implementation, and analyzing the result of the application of the test sequence to determine whether the implementation conforms to the specification.

In a **preset** input sequence, the entire input sequence is fixed prior to applying the first input. In an **adaptive** input sequence, the next input symbol depends on the previously observed outputs.

A **fault model** is defined as a set of models of non-conforming implementations of a given FSM M [IS94]. For a given specification, two types of faults may be recognized:

- **output faults**, i.e., the observed output of an implementation after a specified input may not be the one specified by the specification, and
- **transfer faults**, i.e., the final state of a tested transition of an implementation may be different from the final state specified by the specification.

FSM-based testing adopts a transition-level approach to generating test sequences. If the fault model consists of only output faults, i.e. it is assumed transfer faults do not

occur, then a test sequence that includes every transition of the FSM at least once is sufficient to detect any output faults in the implementation [SB84]. For a fault model which includes transfer faults, in addition to output faults, the following steps must be carried out in order to verify the correct implementation of a transition $t_{jk} = (s_j, s_k; x/y)$:

a. the implementation is transferred to the state recognized as s_j

b. input x is applied and the output produced is checked to match output y as specified by M

c. the state reached as a result of applying x is verified to be s_k.

Steps (b) and (c) together form a **test segment** for transition $t_{jk} = (s_j, s_k; x/y)$.

Clearly, a critical part of transition-level testing is recognizing the starting and terminating states of the transition. This is achieved using input-output sequences specifically designed to determine the present state of the implementation.

Given an FSM M, an input sequence X is a **distinguishing sequence (D)** if the output sequence Y produced by M in response to X is different for each state. $DS(s_i)$ denotes the transition sequence induced by the application of D at state s_i. It is known that a distinguishing sequence may not exist for every minimal FSM [KZ78], and that determining the existence of a distinguishing sequence for an FSM is PSPACE-complete [LY94].

A **Unique input-output (UIO) sequence** for a state s_i, denoted $UIO(s_i)$, is an input/output behaviour that is not exhibited by any other state of the FSM. It is known that UIO sequences may not exist for every state of every minimal FSM [SD88], and determining the existence of a UIO sequence for a state of an FSM is PSPACE-complete [LY94].

A **characterization set** (also called a **W-set**) for an FSM M is a set of input sequences for which the output sequences produced by M in response to this set of input sequences are different for each state of M. A characterization set exists for every minimal FSM [CT78].

Given an FSM M, let $\Phi(M)$ be the set of FSMs each of which has at most n states and the same input and output sets as M. Let M^* be an FSM of $\Phi(M)$. M^* is **isomorphic** to M if there is a one-to-one and onto function f on the state sets of M and M^* such that for any state transition $(s_i, s_j; x/y)$ of M, $(f(s_i), f(s_j); x/y)$ is a transition of M^*. A **checking sequence** of M is an input sequence starting at a specific state of M that distinguishes M from any M^* of $\Phi(M)$ that is not isomorphic to M. In the context of testing, this means that in response to this input sequence, any faulty implementation M^* will produce an output sequence different than the expected output, thereby indicating the presence of a fault.

2.3 Distributed Test Architectures

The application of a test sequence is carried out in a test architecture that consists of testers and the implementation under test (IUT). Both the OSI [II95] and ODP [IS95] models define a number of abstract test architectures.

An **Abstract Test Architecture** (ATA), referred to as an **Abstract Test Method** (ATM) in the OSI model, is a description of how an IUT which is in a **System Under Test** (SUT) is to be tested. An ATA is given at an appropriate level of abstraction to make it independent of any particular implementation, but detailed enough to enable test

cases to be specified in an abstract manner. An ATA for a $2p$-FSM is defined in terms of the relationship between the IUT and the Upper and Lower Testers, through:

Points of Control and Observation (PCO), or points within a testing environment where the occurrence of test events is to be controlled and observed, as defined in an ATA [II95],

Abstract Service Primitives (ASPs), or implementation-independent descriptions of the interactions between a service-user and a service-provider at a service boundary, as defined in an OSI service definition [II95], and

Protocol Data Units (PDUs), or information that is delivered as a unit among peer entities of a network and that may contain control information, address information, or data.

The **Upper Tester** (UT) is defined as the representation of the means of providing, during test execution, control and observation of the upper service boundary of the IUT, as defined by the chosen ATA. The **Lower Tester** (LT) is defined as the representation of the means of providing, during test execution, indirect control and observation of the lower service boundary of the IUT via the underlying service provider. The rules for cooperation among Lower and Upper Testers during testing are defined in the applicable **Test Coordination Procedures** (TCPs) for each architecture.

Within an ATA, an **abstract test suite** (ATS) is defined as a test suite composed of **abstract test cases**, the complete and independent specifications of the actions required to achieve a specific test purpose. The **means of testing** (MOT) is the combination of equipment and procedures that can perform the derivation, selection, parameterization and execution of test cases, in conformance with a reference standardized ATS, and can

produce a conformance log. The **test system** is the real system which includes the realization of the Lower Tester. The same test system can be used as part of several means of testing [II95].

The four OSI ATAs are defined as follows [II95]:

- **Local** - an ATA in which both the Lower and Upper Testers are located within the test system. There are two PCOs, one beneath the Lower Tester and the other at the upper service boundary of the IUT. The upper service boundary is required to be a standardized hardware interface. In the ODP model, this ATA is called **centralized**, which refers to one tester.

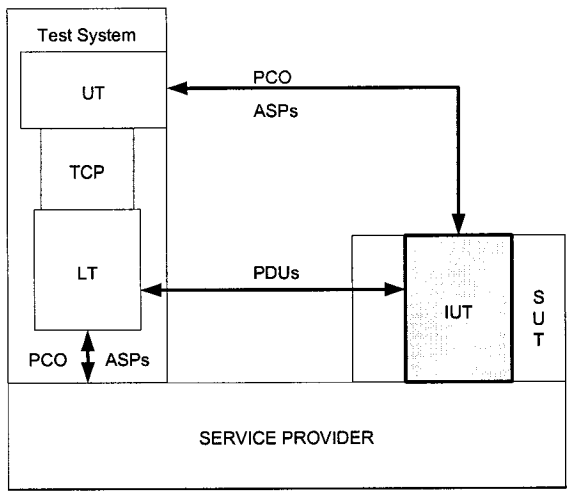


Figure 1. Local Test Architecture

- **Coordinated** – an ATA in which there is one PCO, beneath the Lower Tester. The TCP are realized by means of standardized Test Management Protocols (TMPs), and the Upper Tester is an implementation of the relevant TMP.

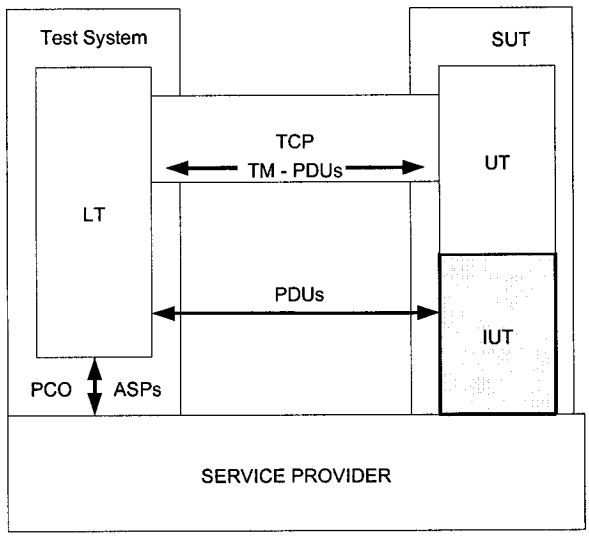


Figure 2. Coordinated Test Architecture

- **Distributed** – an ATA in which the Upper Tester is within the SUT and there are two PCOs, one beneath the Lower Tester and the other at the upper service boundary of the IUT (Figure 3). In this ATA, the upper service boundary must be either a human user interface or a standardized programming language interface.

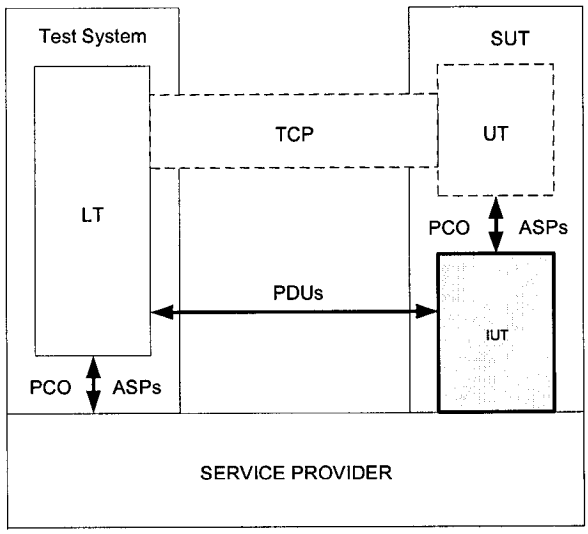


Figure 3. Distributed Test Architecture

- **Remote** – an ATA in which the control and observation of test inputs and outputs is specified solely in terms of Lower Tester activity. Some requirements for TCP may be

implied or informally expressed in the ATS, but no assumption is made regarding their feasibility or realization. There is no Upper Tester as such, but some Upper Tester functions may be performed by the SUT.

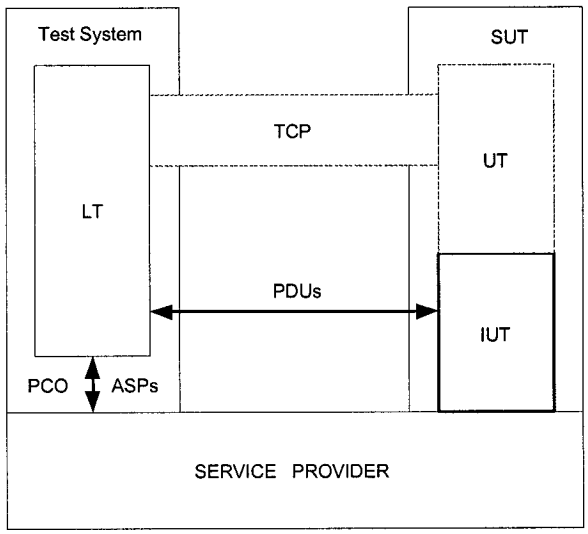


Figure 4. Remote Test Architecture

The ATAs defined by OSI have analogous architectures in other models such as ODP [IS95]. An important difference between the OSI and ODP distributed test architectures is the manner in which testers may communicate during testing. In the OSI architecture, the Upper and Lower testers communicate with one another only indirectly, via their interactions with the IUT. The ODP distributed test architecture, shown in Figure 5, consists of *n* testers located within the test system, with direct communication between testers supported by a multicast channel. External coordination messages exchanged via this channel aid testers in coordinating their respective testing activities.

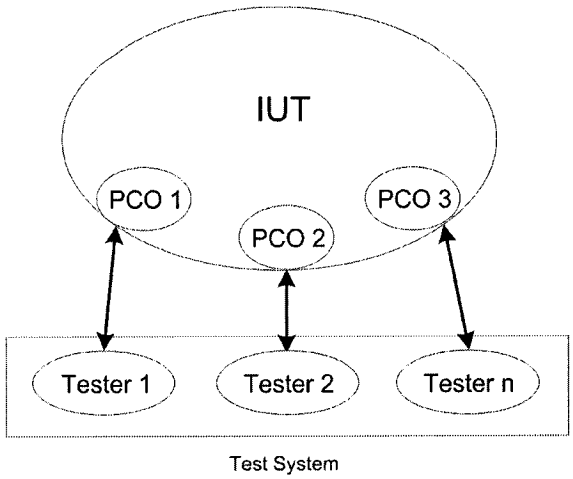


Figure 5. ODP Distributed Test Architecture

A **global test sequence** ω of an *np*-FSM M is the label of a path of the digraph $G = (V, E)$ representing M, which is of the form $x_1/y_1\ x_2/y_2\ \ldots\ x_m/y_m$, where $x_i \in I$ and $y_i \in O$, $1 \leq i \leq m$. Given a global test sequence ω, a **local test sequence** ω_k for port k of an *np*-FSM M is of the form $\alpha_1\ \alpha_2\ \ldots\ \alpha_m$, where α_i $(1 \leq i \leq m)$ is either:

- x_i/y_i, $x_i \in \Sigma_k \cup \{-\}$, $y_i = \langle a_1, a_2, \ldots, a_n \rangle$, where $a_k \in \Gamma_k \cup \{-\}$ and $a_j = -$, for $j \neq k$, $1 \leq j, k \leq n$, or
- the reception or transmission an external coordination message.

Note that $x_i \in \Sigma_k \cup \{-\}$ together with $a_k \in \Gamma_k \cup \{-\}$ allows transitions of the form -/- within a local test sequence ω_k for port k of an *np*-FSM. In an ATA consisting of multiple testers, each tester utilizes a local test sequence constructed from the global test sequence for the given FSM. In the local test sequence, tester k knows only that there are transitions involving itself or others, and does not know the inputs or outputs of the other testers.

The "black box" nature of an IUT allows only limited controllability and observability of the IUT. In this context, **controllability** refers to the ease with which the IUT can be transferred to a designated state and **observability** refers to the ease with

which the current state of the IUT can be recognised. Checking sequence construction methods effectively handle the problems caused by the limited controllability and observability.

Some test architectures may cause additional controllability or observability problems, depending on the ability of the testers to coordinate the application of their respective local test sequences. A **controllability (synchronization) problem** exists when a tester is required to send an input to the IUT in the current transition, and because it is not **involved** in the previous transition, i.e., it did not send the input or receive the output in the previous transition, it does not know when to send the input. An **observability problem** exists when a tester is expecting to receive an output from the IUT in response to either the previous input or the current input, and because it did not send the current input, it does not know when to start or stop waiting for the reception of the output of the IUT.

Given an *np*-FSM M and a global test sequence $\omega = x_1/y_1 x_2/y_2 \ldots x_m/y_m$ of M, where $x_i \in I$ and $y_i \in O$, $1 \leq i \leq m$, a **synchronization problem** occurs when, in the labels x_j/y_j and x_{j+1}/y_{j+1} of any two consecutive transitions, there exists a tester k that sends x_{j+1} that is neither the one sending x_j nor one of those receiving an output belonging to y_j, $1 \leq j \leq m-1$.

Given an *np*-FSM M and a global test sequence $\omega = x_1/y_1 x_2/y_2 \ldots x_m/y_m$ of M, where $x_i \in I$ and $y_i \in O$, $1 \leq i \leq m$, any two consecutive transitions t_{ij} and t_{jk} whose labels are x_j/y_j and x_{j+1}/y_{j+1} form a **synchronizable pair** of transitions if t_{jk} can follow t_{ij} without causing a synchronization problem. For a transition $t_{ij} = (v_i, v_j; x_j/y_j)$, each transition $t_{jk} = (v_j, v_k; x_k/y_k)$ that forms a synchronizable pair of transitions with t_{ij} is called an **eligible successor** of t_{ij}.

Any (sub)sequence of transitions in which every pair of transitions is synchronizable is called a **synchronizable transition (sub)sequence**. A global test sequence is said to be **synchronizable** if it is the label of a synchronizable transition sequence.

Given an *np*-FSM M and a global test sequence $\omega = x_1/y_1\ x_2/y_2\ \ldots\ x_m/y_m$ of M, where $x_i \in I$ and $y_i \in O$, $1 \leq i \leq m$, a **1-shift output fault** in an implementation N of M exists when, in the labels x_j/y_j and x_{j+1}/y_{j+1} of any two consecutive transitions, there exists one a_k in $y_j = \langle a_1, a_2, \ldots, a_n \rangle$ of M which occurs in $y_{j+1} = \langle a_1, a_2, \ldots, a_n \rangle$ in N (and not in y_j in N) or there exists one a_k in $y_{j+1} = \langle a_1, a_2, \ldots, a_n \rangle$ of M which occurs in $y_j = \langle a_1, a_2, \ldots, a_n \rangle$ in N (and not in y_{j+1} in N), $1 \leq j \leq m\text{-}1$, $1 \leq k \leq m$. An instance of the observability problem manifests itself as an **undetectable 1-shift output fault** if there is a 1-shift output fault related to $a_k \in \Gamma_k$ in any two consecutive transitions whose labels are x_j/y_j and x_{j+1}/y_{j+1}, such that tester k satisfies the condition (a_k is in y_j XOR a_k is in y_{j+1}) AND $x_{j+1} \notin \Sigma_k$. In this case, we say that tester k is **involved** in the shift, and would not be able to detect it.

Chapter Three

Previous Work

3.1 Checking Sequence Generation

The existing literature on FSM-based testing proposes several methods for the construction of checking sequences for the local test architecture. These methods typically assume that for an n-state specification FSM M, the implementation does not change during execution and has at most n distinct states. Several proposed methods assume the presence of a reset input that can be applied at any state and takes the implementation to the initial state s_1. Solutions employing a reliable reset consist of two parts; a **state cover**, composed of subsequences which verify the uniqueness of every state of M in the implementation, and a **transition cover**, whose subsequences verify the correctness of every transition (s_i, s_j; x/y) of M in the implementation.

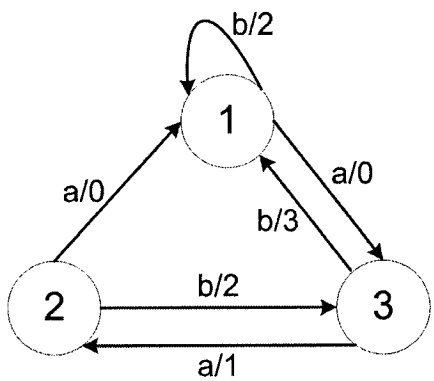


Figure 6. Digraph $G = (V, E)$ of FSM $M1$

A preamble for a state s_i, denoted preamble(s_i), is the transition sequence represented by the shortest path from v_1 to v_i on $G = (V, E)$. If a distinguishing sequence is used, the state cover consists of a transition sequence $preamble(s_i)@DS(s_i)$ for each state s_i. Consider the FSM $M1$ shown in Figure 6, which has a distinguishing sequence $D = ab$.

For each transition sequence (which is a path) of the state cover for $M1$, the input portion of the label of each sequence is shown in Table 1. For readability, the concatenation symbol @ will not be shown within an actual sequence hereafter.

Table 1. State cover for FSM $M1$

State	Input Sequence
s_1	*ab*
s_2	*aaab*
s_3	*aab*

For each transition (s_i, s_j; *x/y*), a transition sequence *preamble*(s_i)@(s_i, s_j; *x/y*)@*DS*(s_j) is created. These sequences together form the transition cover. For each of the transition sequences forming the transition cover for $M1$, the input portion of the labels of these sequences are shown in Table 2.

Table 2. Transition cover for FSM $M1$

Transition	Input Sequence
(s_1, s_2; *a/0*)	*aab*
(s_1, s_1; *b/2*)	*bab*
(s_2, s_1; *a/0*)	*aaaab*
(s_2, s_3; *b/2*)	*aabab*
(s_3, s_2; *a/1*)	*aaab*
(s_3, s_1; *b/3*)	*abab*

Prefix elimination removes redundant transition sequences from the state and transition covers; if a sequence S_P is a prefix of some sequence S_Q , then S_P is eliminated. Input portions of the labels of the sequences that remain following this step form the checking sequence. Applying this approach to FSM $M1$ eliminates 4 sequences and results in the checking sequence:

r bab r aaaab r aabab r aaab r abab

Similar reset-based approaches have been proposed for methods in which states are recognised using UIO sequences [SD88] and characterization sets [CT78, FB91].

Several methods have been proposed for testing implementations which do not implement a reset function, or for which a reset is a costly transition. One approach is to replace the reset inputs in the above method with an input sequence that takes the implementation to the initial state s_1. This can be accomplished using a preset synchronizing sequence if one exists; otherwise, adaptive input sequences can be used.

Hennie [HF64] exploits similarities among input-output sequences to recognise states and thereby reduce the overall length of the checking sequence. The method employs a distinguishing sequence D. Let $G = (V, E)$ be the graphical representation of FSM M, and let the input-output sequence Q be the label of a path P of G. A state s_r represented by n_r in G is ***d*-recognized** in Q if Q contains the subpath $D/\lambda(n_r, D)$.

Suppose that $(n_q, n_i; T)$ and $(n_j, n_k; T)$ are subpaths of P and $D/\lambda(a, D)$ is a prefix of T; that is, both n_q and n_j are recognized as some state a of M. Now suppose that node n_k is d-recognized in Q as some state a' of M. Then node n_i is said to be ***t*-recognized** in Q as state a' of M (Figure 7a). Now suppose that $(n_q, n_i; T)$ and $(n_j, n_k; T)$ are subpaths of P where nodes n_q and n_j are either d-recognized or t-recognized in Q as some state a of M, and node n_k is either d-recognized or t-recognized in Q as some state a' of M. Then node is t-recognized in Q as state a' of M (Figure 7b). If a node n is d-recognized or t-recognized as a state s then it is said to be **recognized** as state s.

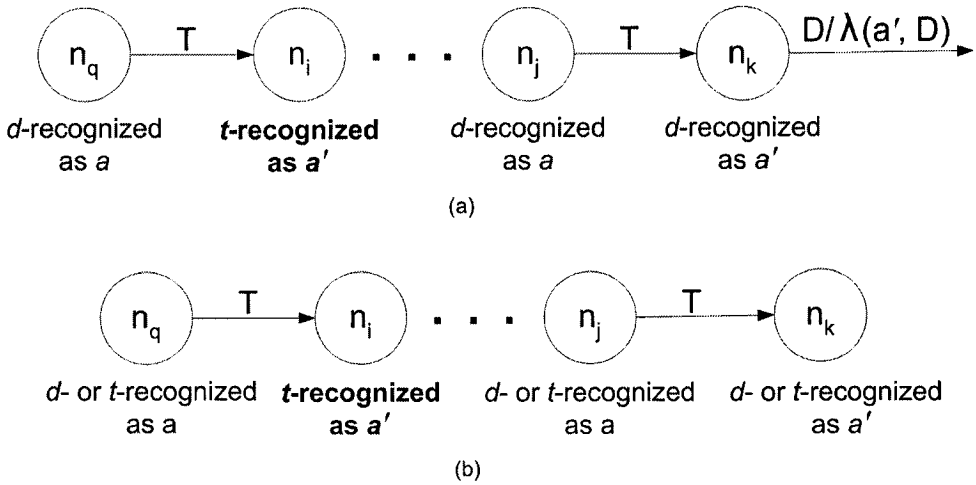


Figure 7. *t*-recognition of s_i

The method presented in [HF64] first generates an input-output sequence that applies the distinguishing sequence D at each state. Let $q_i = \delta(s_i, D)$. A **transfer sequence** $T(s_i, s_j)$ is an input sequence that transfers the FSM from s_i to s_j. It is assumed the implementation has been brought to its initial state s_1. The input sequence D is applied, bringing the implementation to some state q_1. The transfer sequence $T(q_1, s_2)$ is then applied, which brings the implementation to state s_2. D is applied at s_2, followed by $T(q_2, s_3)$. This procedure continues until D has been applied at all n states. Finally, $T(q_n, s_1)$ is applied and D is again applied at initial state, resulting in the sequence:

$D\ T(q_1, s_2)\ D\ T(q_2, s_3)\ D\ \ldots\ D\ T(q_{n-1}, s_n)\ D\ T(q_n, s_1)\ D$

The second phase of the method verifies the transitions in a breadth-first order within the order of states from s_1 to s_n. Note that after applying the sequence above, a correct implementation will be in a state recognized as q_1. The transition $(s_1, s_k; x/y)$ is verified by applying the sequence:

$T(q_1, s_n)\ D\ T(q_n, s_1)\ x\ D$

In this sequence, $T(q_1, s_n)$ transfers the implementation to s_n, which is verified by applying D. The same transfer sequence $T(q_n, s_1)$ used in the first phase is again applied,

which allows us to t-recognize s_1 as $head(s_1, s_k; x/y)$. The input x of $(s_1, s_k; x/y)$ is applied, and the corresponding output produced is checked to match y. D is then applied to verify $tail(s_1, s_k; x/y)$ as s_k. This procedure is repeated to verify each transition; i.e. if the implementation is in state q_i and the next transition to be verified is $(s_j, s_k; x/y)$, the sequence applied is:

$T(q_i, s_{j-1})\ D\ T(q_{j-1}, s_j)\ x\ D$

Concatenating the sequence generated in phase 1 and the sequence generated in phase 2 completes the construction of the checking sequence.

Gonenc [GG70] proposes improvements to the method in [HF64] which can reduce the lengths of both the sequence generated in phase 1, referred to in [GG70] as the **α-sequence,** and the sequence generated in phase 2, referred to as the **β-sequence**. The α-sequence is generated by first constructing an **(X_d)-diagram** representing the application of D at each state. For each state s_i, a vertex i is created, and an edge $(i, j; D/\lambda(s_i, D))$ added to the (X_d)-diagram, where $s_j = \delta(s_i, D)$ and for state s_j a vertex j is created. A source node, if any exist in the (X_d)-diagram, is chosen as the starting state. Otherwise an arbitrary node is chosen. The following algorithm is then applied:

(1) *Apply D.*

(2) *If the current state is not yet recognized, go to* (1). *Else go to* (3).

(3) *Apply D.*

(4) *If there is a source node not yet recognized, go to* (5). *Else go to* (6).

(5) *Apply $T(s_i, s_j)$ where s_i is the current state and s_j is (one of) the source node(s) not yet recognized. Then go to* (1).

(6) *If there remain any unrecognized states, go to* (7). *Else Stop.*

(7) *Apply $T(s_m, s_n)$ where s_m is the current state and s_n is the state not yet recognized. Then go to* (1).

The algorithm ensures that all states are recognized, and the final state of the α-sequence is t-recognized. This approach may generate both fewer and shorter transfer sequences than those of [HF64] as the states are not required to be verified in a specific order.

The **β-sequence** is constructed using a **β-diagram** whose edges represent transition sequences which verify each transition. For each state s_i, a vertex i is created. For each transition $(s_i, s_j; x/y)$, a subsequence xD is created and the corresponding edge $(i, k; xD/\lambda(xD))$ added to the β-diagram, where $s_k = \delta(s_i, xD)$ and the vertex k stands for state s_k. The β-sequence is then generated using the β-diagram, beginning at the state at which the α-sequence terminated. Edges of the β-diagram are removed as their corresponding input sequences are added to the β-sequence. The β-sequence is complete when no edges remain. If a node i in the β-diagram is reached which has no outgoing edges, the shortest transfer sequence $T(s_i, s_j)$ is applied, where s_j is the nearest state that has an outgoing edge in the β-diagram. This approach is an improvement upon [HF64] as the transitions are not required to be verified in a specific order. The checking sequence generated in [GG70] consists of the α-sequence followed by the β-sequence.

Ural, Wu, and Zhang [UW97] present further improvements to the methods of [HF64] and [GG70]. A set of **α-sequences** is introduced, and a digraph $G' = (V', E')$ is obtained from the given digraph $G = (V, E)$. Edges in E' include a set of edges (E_α) representing the α-sequences, and a set of edges (E_C) that verify each transition. The

input portion of the label of an RPP over the edges $E_\alpha \cup E_C$ in G' forms a checking sequence.

The construction of E_α is facilitated by forming of a set of paths $P_1, \ldots, P_q$ of G where each path P_k induces an edge of E_α whose label is the *α-sequence* α_k, $1 \leq k \leq q$. Specifically,

a) the set of vertices $V_k \subseteq V$ covered by P_k, $1 \leq k \leq q$, is $\{v_1^k, v_2^k, \ldots, v_{m_k}^k\}$;

b) the union of the V_k is V.

c) the label of P_k, α-sequence α_k, is $D/\lambda(v_1^k, D)T_1^k \;\; D/\lambda(v_2^k, D)T_2^k \;\; \ldots \;\; D/\lambda(v_{m_k}^k, D)T_{m_k}^k$

where for $1 \leq j \leq m_k$, $T_j^k = (X_j^k / Y_j^k)$ is a (possibly empty) transfer sequence from $\delta(v_j^k, D)$ to v_{j+1}^k, $v_{m_k+1}^k = v_w^k$, and v_w^k is any member of V_k.

When a path P_k, $label(P_k) = \alpha_k$, $1 \leq k \leq q$, is contained in a path P of G then

a) v_j^k, $1 \leq j \leq m_k$, is *d*-recognized in α_k;

b) $\delta(v_j^k, DX_j^k)$, $1 \leq j \leq m_k$, is *d*-recognized in α_k; and

c) $tail(P_k)$ is recognized in α_k.

From the elements of α-set, a set of transfer sequences, called ***T-set,*** is formed as a set of labels of subpaths $R_1, \ldots, R_p$ of paths $P_1, \ldots, P_q$, such that each element T_i of *T-set* is label(R_i) where $\{R_i: i = 1, 2, \ldots, p\} = \{(v_j^k, \delta(v_j^k, DX_j^k);\ D/\lambda(v_j^k, D)T_j^k): 1 \leq k \leq q \text{ and } 1 \leq j \leq m_k\}$. Note that $head(R_i)$ is recognized because D is applied to $head(R_i)$, and $tail(R_i)$ is recognised in some α_k because $tail(R_i)$ is $\delta(v_j^k, DX_j^k)$, to which D is applied.

The auxiliary graph $G' = (V', E')$ is constructed by first creating vertices $V' = V \cup U'$ where $U' = \{v_i'$: for every $v_i \in V\}$. The set of paths $P_1, \ldots, P_q$ and set of subpaths $R_1, \ldots,$

R_p are included in E' as edges in $E_\alpha = \{(head(P_k), (tail(P_k))'; \alpha_k): 1 \leq k \leq q\}$ and $E_T = \{($ $head(R_i), (tail(R_i))'; T_i): 1 \leq i \leq p\}$, respectively. These edges facilitate the recognition of vertices in the label Q of a path P' of G'. A test segment for each edge $(v_i, v_j; x/y)$ of G is added to E' as edges in $E_C = \{(v'_i, (\delta(v_i, xDX_j^k))'; (xDX_j^k)/\lambda(xDX_j^k)): (v_i, v_j; x/y) \in E\}$. Finally, two sets of edges $E_\epsilon \subset E'$ and $E'' \subset E'$ are added to increase the connectivity of G'; $E_\epsilon = \{(v'_i, v_i; \epsilon): v_i \in V\}$ and E'' is a subset of $\{(v'_i, v'_j; x/y): (v_i, v_j; x/y) \in E\}$ such that $G'' = (U', E'')$ has no tour and G' is strongly connected.

Once G' is formed, an RPP P' of G' is found that contains all edges in $E_\alpha \cup E_C$. Since G' is obtained from G, P' represents a path P of G. In [UW97] the RPP is generated by first finding the minimal symmetric augmentation G'' of $(V', E_\alpha \cup E_C)$, which is produced by adding edges from E' to G'. If G'', with its isolated vertices removed, is connected, G'' has an Euler tour and this forms P'; otherwise, a heuristic is applied to make G'' connected and an Euler tour is formed. If G'' is connected, P' is an RCPP over $E_\alpha \cup E_C$. The input portion of the sequence represented by the label of P' is a checking sequence. The authors show that the methods in [GG70] and [HF64] are special cases of their own, and provide a formal proof that their method generates a checking sequence.

Hierons and Ural [HU02a] make two refinements to the method in [UW97] that further reduce the length of the checking sequence produced. First, α'-sequences are introduced, which are similar to α-sequences but are not required to end within their own body. Stated formally, an **α'-sequence** $\alpha'_k = D/\lambda(v_1^k, D)T_1^k \; D/\lambda(v_2^k, D)T_2^k \ldots D/\lambda(v_{m_k}^k, D)T_{m_k}^k \; D/\lambda(v_w^{k'}, D)T_w^{k'}$ where $T_j^k = (X_j^k / Y_j^k)$ is a (possibly empty) transfer sequence from $\delta(v_j^k, D)$ to v_{j+1}^k for $1 \leq j \leq m_k$, $v_{m_k+1}^k = v_w^{k'}$,

and $v_w^{k'}$ is contained in **any** $V_{k'}$, $1 \le k' \le q$ and $1 \le w \le m_{k'}$. Thus $tail(P_k)$, $\alpha'_k = label(P_k)$, is recognized in some $\alpha'_{k'}$, $1 \le k' \le q$. Note also that every α_k is an α'_k but the converse is not true.

The second modification presented in [HU02a] exploits the property that every $label(P_k)$ and every $label(R_i)$ begins with the distinguishing sequence. Thus an α'_k or T_i can be used to verify the end state of a transition in forming a test segment for that transition. Therefore the edges E_C in [UW97] are replaced by $E_C = \{(v'_i, v_j; x/y): (v_i, v_j; x/y) \in E\}$. The edge sets E and E_ϵ included in [UW97] are also removed as the connectivity they provided is already guaranteed. The set of edges E_T is formed as in [UW97], and so $E' = E_\alpha \cup E_T \cup E_C \cup E''$.

Consider the 2p-FSM $M2$ from [HU02a] shown in Figure 8, with $D = aba$. An α'-set for $M2$ is $\{\alpha'_1, \alpha'_2\}$ where α'_1 , the label of $P'_1 = (s_5, s_4; \alpha'_1)$, is $D/\lambda(s_5, D)$ $D/\lambda(s_2, D)$ $D/\lambda(s_4, D)$ $D/\lambda(s_1, D)$ $D/\lambda(s_2, D)$ and α'_2, the label of $P'_2 = (s_3, s_2; \alpha'_2)$, is $D/\lambda(s_3, D)$ $D/\lambda(s_1, D)$. Then in G', $E_\alpha = \{(v_5, v'_4; \alpha'_1), (v_3, v'_2; \alpha'_2)\}$ and $E_T = \{(v_1, v'_2; T_1), (v_2, v'_4; T_2), (v_3, v'_1; T_3), (v_4, v'_1; T_4), (v_5, v'_2; T_5)\}$. E'' consists of two edges $(v'_2, v'_5; a/x)$ and $(v'_5, v'_3; b/y)$ which make G' connected. The resulting graph G' is shown in Figure 9.

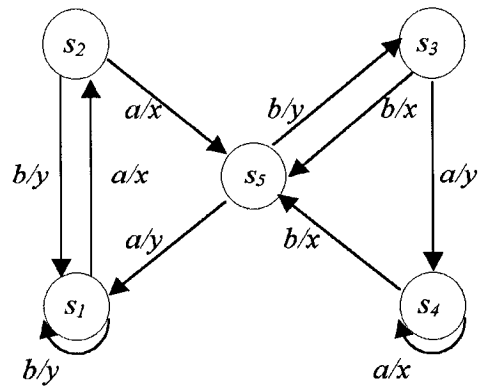


Figure 8. Digraph $G = (V, E)$ of 2p-FSM $M2$

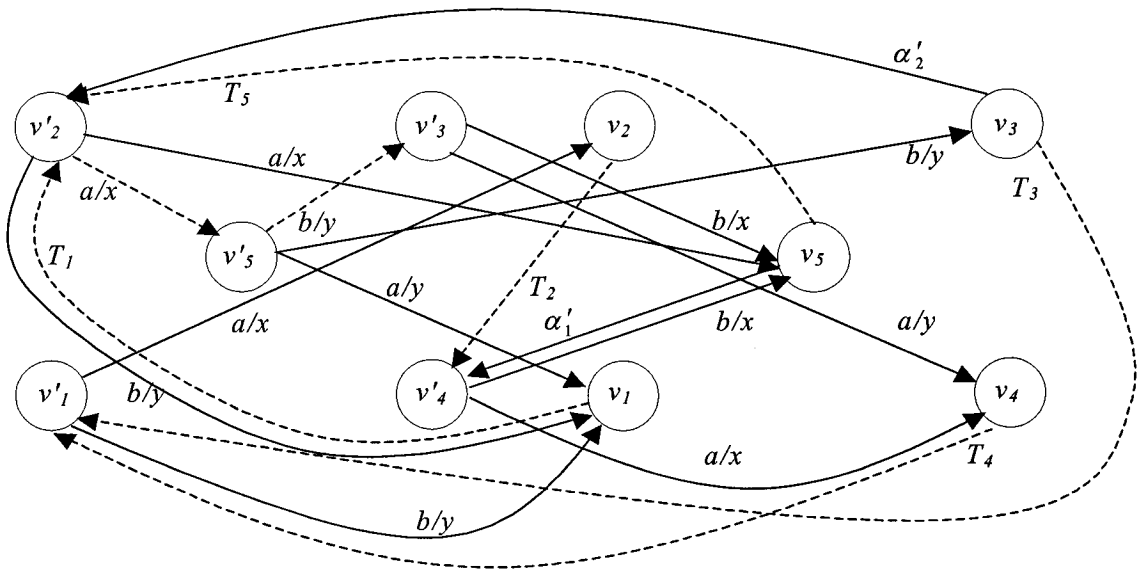


Figure 9. Digraph $G' = (V', E')$ of 2p-FSM $M2$

Similar to [UW97], an RPP over the edges $E_\alpha \cup E_C$ is found. The minimal symmetric augmentation G'' of the edge set $E_\alpha \cup E_C$, shown for 2p-FSM $M2$ in Figure 10, is formed by adding edges from G'.

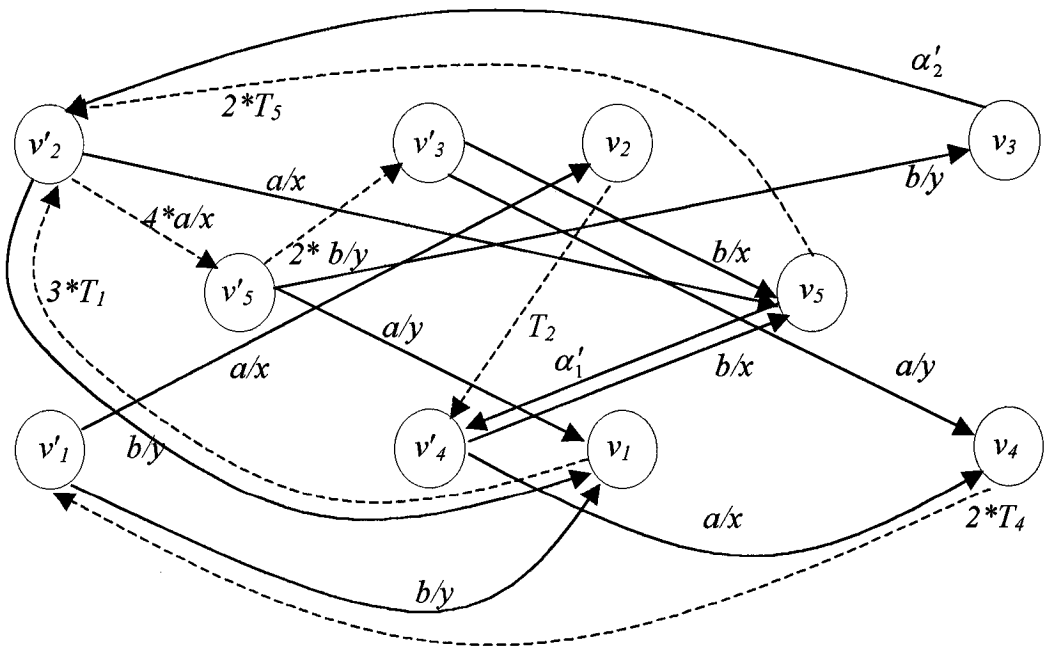


Figure 10. Digraph $G'' = (V'', E'')$ of 2p-FSM $M2$

An Euler tour of G'' produces a checking sequence. The checking sequence generated for $M2$ from G'' in Figure 10 is the input portion of:

b/y, $D/\lambda(s_1,D)$, b/y, $D/\lambda(s_1,D)$, a/x, α'_1, a/x, $D/\lambda(s_4,D)$, a/x, $D/\lambda(s_2,D)$, b/x, $D/\lambda(s_5,D)$, a/x, b/y, α'_2, a/x, a/y, $D/\lambda(s_1,D)$, a/x, b/y, b/x, $D/\lambda(s_5,D)$, a/x, b/y, a/y, $D/\lambda(s_4,D)$

The authors of [HU02a] demonstrate that their two proposed enhancements generate significantly shorter checking sequences than the method of [UW97]. They also suggest two possible improvements, the first being to choose the connecting transitions from the transitions of the given FSM during optimization, as opposed to choosing from the subset E'' found prior to optimization. Secondly, only prefixes of D might be used in recognizing states [HU02a].

3.2 Controllability (Synchronization) Problem and Solutions

Recall from Section 2.3 that a **controllability (synchronization) problem** exists when a tester is required to send an input to the IUT in the current transition, and because this tester did not send the input or receive output in the previous transition, it does not know when to send the input. A global test sequence is **synchronizable** if it is the label of a synchronizable transition sequence, i.e. it contains no synchronization problems.

Sarikaya and Bochman [SB84] were the first to formally address the synchronization problem and propose a method for constructing synchronizable global test sequences. In their approach, each transition or subsequence to be added to the global test sequence is checked to determine whether it is synchronizable with its predecessor. If it is not synchronizable, a different transition or subsequence from the present state is considered. If no synchronizable transition from the present state exists, the algorithm backtracks

until either a synchronizable global test sequence is found or it is determined that no such sequence exists for the given FSM. The authors note that this approach may result in a global test sequence that is not of minimum length. In addition, a protocol specification is **intrinsically nonsynchronizable** if it contains a transition that does not have an eligible successor. The authors propose a procedure for intrinsically nonsynchronizable protocols in which the protocol specification is modified, for the purpose of testing, so that a synchronizable global test sequence can be found.

Boyd and Ural [BU91] give a necessary and sufficient condition for the existence of a synchronizable global test sequence for a given FSM that can be checked in polynomial time. They also show that it is unlikely that a polynomial time algorithm will be found to obtain a minimum length synchronizable global test sequence from a given $2p$-FSM in the general case. Their work provides justification for heuristic techniques for generating synchronizable global test sequences which are not necessarily of minimum length.

The methods proposed in [UW93, CU95, GU95] are based on such heuristic techniques for the solution of the synchronization problem. They construct an auxiliary graph from the given FSM and a tour of this auxiliary graph yields a synchronizable global test sequence [UW93, CU95] or checking sequence [GU95]. These methods require two synchronizable unique input-output sequences (SUIOs) for each state s, denoted $\text{SUIO}^U(s)$ and $\text{SUIO}^L(s)$, with the first input from Upper Tester and Lower Tester respectively. Synchronization problems between a UIO and its predecessor are thereby eliminated, as an SUIO that is an eligible successor is used. A digraph $G = (V, E)$ is said to be **order-specified** if for each edge $e_{ij} \in E$, a subset of the edges leaving vertex v_j is specified as eligible successors of e_{ij}, and a path in G is said to be **correctly ordered** if

for every consecutive pair of edges $e_{jk} = (v_j, v_k; x_j/y_j)$ and $e_{kl} = (v_k, v_l; x_k/y_k)$, e_{kl} is an eligible successor of e_{jk}. Given a $2p$-FSM M represented by an order-specified digraph $G = (V, E)$, [UW93] first constructs a duplex digraph $G' = (V', E')$. Test segments are constructed using the SUIOs, and edges representing these subsequences are added to G' to form $G^{\wedge} = (V^{\wedge}, E^{\wedge})$. An RPT in $G^{\wedge}$ over the set of edges representing test segments is then found, which represents a synchronizable test sequence for the given FSM.

The method proposed [GU95] assumes the existence of a reliable reset feature and aims at constructing a checking sequence. The solution is based on the construction of a correctly-ordered digraph $G' = (V', E')$ such that all edges of G are in one to one correspondence with edges in G', and all paths in G' are correctly ordered paths in G. SUIO sequences are used to construct test subsequences for a state cover and a transition cover. Each of these test subsequences start at the initial state and are connected by a reset input. The checking sequence constructed by this method is minimized by eliminating redundant subsequences and the judicious choice of transition sequences and SUIOs.

The methods described above make the assumption that the testers may only communicate with each other indirectly via their interactions with the IUT. It is recognized that these methods will not yield synchronizable global input-output sequences for all FSMs because some FSMs are intrinsically nonsynchronizable. The method presented in [CU95] supports an additional communication channel that facilitates external coordination message exchanges related to controllability amongst the testers, and the cost of these messages are included in the minimization algorithm. The use of such external coordination messages allows for the construction of a

synchronizable global test sequence for any FSM, including those which are intrinsically nonsynchronizable. In [CU95] a digraph $G' = (V', E')$ is constructed by adding the set of test segments, denoted E_C, to $G = (V, E)$, and finding an RCPT of G' over E_C. It is shown that if the edge induced spanning subgraph $G[E_C]$ of G' is weakly connected, the RCPT of G' over E_C can be found in polynomial time.

The solution proposed in [CR99] also uses external coordination messages to resolve controllability problems. Recall from Section 2.3 that, given an *np*-FSM M and a global test sequence $\omega = x_1/y_1, x_2/y_2, \ldots, x_m/y_m$ of M, where $x_i \in I$ and $y_i \in O$, $1 \leq i \leq m$, a **synchronization problem** occurs when, in the labels x_j/y_j and x_{j+1}/y_{j+1} of any two consecutive transitions, there exists a tester k that sends x_{j+1} that is neither the one sending x_j nor one of those receiving an output belonging to y_j , $1 \leq j \leq m$-1. In the method proposed in [CR99], tester h can be the tester that receives an output from the IUT in y_j or, in the case where there exists no output from the IUT in y_j, tester h is the tester that sends an input to the IUT in x_j. For the purposes of this explanation we will assume tester h is the tester sending an input to the IUT in x_j.

To resolve synchronization problems, [CR99] proposes the insertion an external coordination message exchange "<-C_k, +C_h>" relating to controllability between x_j/y_j and x_{j+1}/y_{j+1} in the global test sequence, which corresponds to the insertion of:

- "-C_k" in the local test sequence of tester h just after sending the input x_j to the IUT to indicate that it is to send an external coordination message to tester k, and
- "+C_h" in the local test sequence of tester k just before sending the input x_{j+1} to the IUT, to indicate that it is to receive an external coordination message from tester h telling tester k it is time to send its input to the IUT.

Consider the 2*p*-FSM *M*3 from [CR99] shown in Figure 11. Henceforth, Upper Tester will be denoted *U*, and Lower Tester will be denoted *L*. The authors assume only output faults, and start with a transition tour of the graph *G* representing *M*. One such tour is *t*1, *t*2, *t*3, *t*4. The label of this sequence, (*a*/<0, ->, *a*/<0, ->, *b*/<-, 1>, *b*/<0, ->), contains a synchronization problem between transitions *t*2 and *t*3; in transition *t*3, the Lower Tester is required to send an input *b* to the IUT, but is not involved in transition *t*2, and therefore does not know when to send this input. Also note that *t*3 causes a synchronization problem whether it follows *t*1 or *t*2, and so there is no synchronizable global test sequence for this FSM that does not require external coordination between testers.

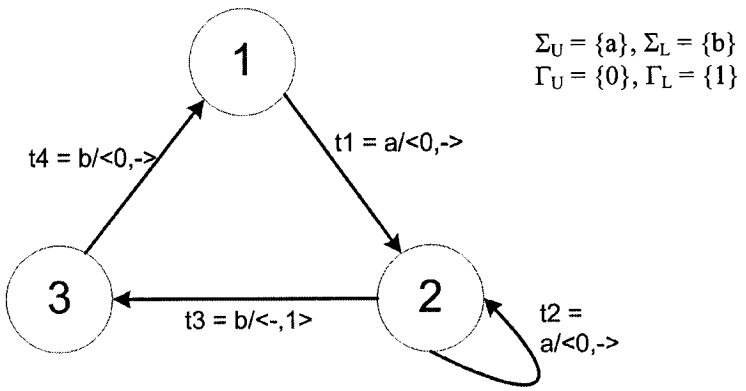


Figure 11. Digraph *G* = (*V*, *E*) of 2*p*-FSM *M*3

The synchronization problem that occurs between transitions *a*/<0, -> and *b*/<-, 1> in the sequence above is resolved by inserting an external coordination message exchange <-C_L, +C_U> between *a*/<0, -> and *b*/<-, 1> in the global test sequence. Thus, the global test sequence ω of *M*3, which is:

a/<0, ->, *a*/<0, ->, *b*/<-, 1>, *b*/<0, ->, becomes

a/<0, ->, *a*/<0, ->, <-C_L, +C_U>, *b*/<-, 1>, *b*/<0, ->,

from which the local test sequences are derived for the Upper and Lower Tester, respectively:

- $\omega_U = a/<0, ->, a/<0, ->, -C_L, -/<-, ->, -/<0, ->$
- $\omega_L = -/<-, ->, -/<-, ->, +C_U, b/<-, 1>, b/<-, ->$

Tai and Young [TY98] introduce an alternative definition of synchronizable which depends on the current state and eligible transitions of the FSM as well as the transitions in the local test sequences. Specifically, their definition states that a test sequence is synchronizable if any execution of M and the testers according to the global test sequence is deterministic (i.e., the current state of M has at most one eligible transition). Tai and Young refer to the synchronization problem as it is defined in this thesis as the **pair-wise synchronization problem**. The authors also discuss both port-based testing, which allows testers to communicate with each other only indirectly via their interactions with the IUT, and group-based testing, which divides the ports of an IUT into groups and allows the testers for ports in the same group to communicate with each other directly using external coordination messages.

3.3 Observability Problem and Solutions

Recall from Section 2.3 that an **observability problem** exists when a tester is expecting to receive an output from the IUT in response to either the previous input or the current input, and because it did not send the current input, it does not know when to start or stop waiting for the reception of the output of the IUT. Solutions that assume only indirect communication among testers generate additional synchronizable subsequences for the purpose of resolving observability problems. Those solutions which support

direct communication resolve such problems by inserting external coordination message exchanges relating to observability into the global test sequence.

In [YT98], Young and Tai describe three types of implementation faults that cause incorrect test observations. Given an FSM M, an implementation N of M, and a global test sequence ω, N is said to have an **input exchange fault** with respect to ω if, for two transitions t_{ij} and t_{kl} in ω, N is modified from M by exchanging the input symbols of t_{ij} and t_{kl}. N is said to have a **forward output shifting fault** with respect to ω if N is modified from M by removing an output symbol from t_{ij} and adding it to t_{kl}, provided the input symbol of t_{kl} is associated with a different port than the port of the output being shifted. N is said to have a **backward output shifting fault** with respect to ω if N is modified from M by removing an output symbol from t_{kl} and adding it to t_{ij}, provided the input symbol of t_{kl} is associated with a different port than the port of the output being shifted. The authors assume that testers may communicate only indirectly via their interactions with the IUT. Their strategy to solving the observability problem is to validate the transitions in the global test sequence one at a time, i.e.: given a global test sequence consisting of transitions $t1, t2, t3, \ldots, tn$ for a given FSM M and the implementation N of M, first test N by using $t1$. If this test is successful, test N using $t1$, $t2$, then $t1$, $t2$, $t3$, and so on.

The observability problem is defined by [LB94] as follows: for two adjacent transitions t_{jk} and t_{kl} in a given specification M, a faulty implementation N can be obtained from M by removing an output from one of these two transitions and adding the output to the other transition. [LB94] calls this class of output faults **output shifting faults** (referred to in this thesis as 1-shift output faults), and proposes an approach that adds specific transitions to the synchronizable global test sequence, with the intent of detecting

all 1-shift output faults during the application of the sequence in the distributed test architecture. [LB94] proposes the following algorithm to accomplish this:

- generate a synchronizable global test sequence Π by using one of the test generation methods for FSMs,
- find a set Ω of all consecutive transition pairs along the path caused by applying Π, where each pair may have a potential undetectable 1-shift output fault,
- if Ω is empty, stop. Otherwise, add a set of additional synchronizable test subsequences to Π such that Π can ensure the absence of potential undetectable 1-shift output faults in the transition pairs of Π.

Consider the 2*p*-FSM *M*4 in Figure 12. A synchronizable global test sequence for *M*4 is Π = *label*(*t*1, *t*3, *t*2, *t*4), i.e., *a*/<0, 1>, *b*/<-, 2>, *b*/<0, ->, *a*/<0, ->. Two potential undetectable 1-shift output faults exist in this sequence: a forward shift of "0" from *t*1 to *t*3, and a backward shift of "0" from *t*2 to *t*3.

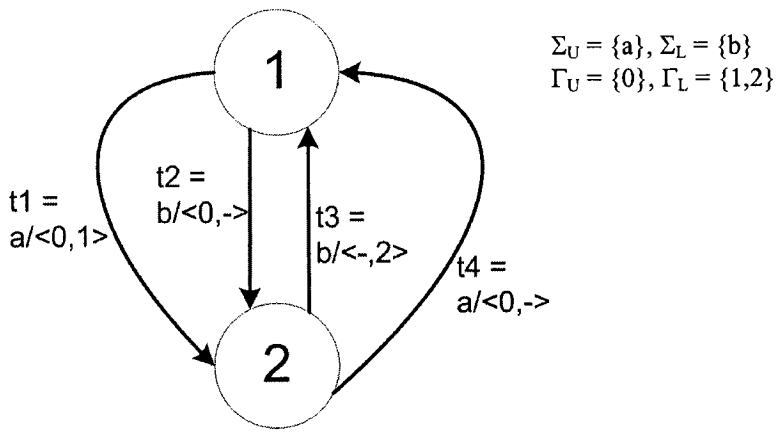


Figure 12. Digraph *G* = (*V*, *E*) of 2*p*-FSM *M*4

These problems are resolved by two additional subsequences:

- *a*/<0, 1>, *a*/<0, -> will detect a 1-shift output fault from *t*1 to *t*3. This is because the Upper Tester will not send the input "*a*" in transition *a*/<0, -> until it receives the

output "0" in transition *a*/<0, 1>. If that output has shifted, the Upper Tester will not receive "0" in response to "*a*" and therefore will detect the shift.

- *a*/<0, 1>, *b*/<-, 2> will detect a 1-shift output fault from $t2$ to $t3$. This is because the Upper Tester is not expecting the second output "0" in the incorrect transition *b*/<0, 2> from the IUT after it sends the input "*a*". If that output has shifted, and given that *b*/<-, 2> is the last transition in the appended synchronizable global test sequence, the Upper Tester will therefore detect the shift.

Thus, the new synchronizable global test sequence is:

a/<0, 1>, *b*/<-, 2>, *b*/<0, ->, *a*/<0, ->, *a*/<0, 1>, *a*/<0, ->, *a*/<0, 1>, *b*/<-, 2>,

in which the potential undetectable 1-shift output faults in the original synchronizable global test sequence are rendered detectable. Note that the appended subsequences introduce three additional potential undetectable 1-shift output faults, all involving the shift of output "1" from *a*/<0, 1> to *a*/<0, ->. However, these potential undetectable 1-shift output faults are rendered detectable by the transition pair *a*/<0, 1>, *b*/<-, 2> in the original synchronizable global test sequence. The authors note that for some *np*-FSMs, full fault coverage cannot be ensured by the method as synchronizable subsequences required to resolve observability problems may not exist [LB94].

Hierons and Ural [HU02b] propose a method for generating a checking sequence based on UIO sequences. It is assumed in [HU02b] that testers may communicate only indirectly via their interactions with the IUT. To verify the uniqueness of state s_i, the input portion of the UIO sequence for every state must be applied at s_i. Therefore, the method requires that the input portion of every UIO sequence can be applied at every state without causing synchronization problems. Potential 1-shift output faults are

detected by adding specific transitions to the synchronizable checking sequence, as in [LB94].

Recall from Section 2.3 that, given an *np*-FSM M and a global test sequence $\omega = x_1/y_1, x_2/y_2, \ldots, x_m/y_m$ of M, where $x_i \in I$ and $y_i \in O$, $1 \leq i \leq m$, a **1-shift output fault** in an implementation N of M exists when, in the labels x_j/y_j and x_{j+1}/y_{j+1} of any two consecutive transitions, there exists one a_k in $y_j = <a_1, a_2, \ldots, a_n>$ of M which occurs in $y_{j+1} = <a_1, a_2, \ldots, a_n>$ in N (and not in y_j in N), or there exists one a_k in $y_{j+1} = <a_1, a_2, \ldots, a_n>$ of M which occurs in $y_j = <a_1, a_2, \ldots, a_n>$ in N (and not in y_{j+1} in N), $1 \leq j \leq m$-1, $1 \leq k \leq m$. An instance of the observability problem manifests itself as an **undetectable 1-shift output fault** if there is a 1-shift output fault related to $a_k \in \Gamma_k$ in any two consecutive transitions whose labels are x_j/y_j and x_{j+1}/y_{j+1}, such that tester k satisfies the condition (a_k is in y_j XOR a_k is in y_{j+1}) AND $x_{j+1} \notin \Sigma_k$. In this case, we say that tester k is **involved** in the shift, and would not be able to detect it.

As in their solution to controllability problems examined in Section 3.2, Cacciari and Rafiq [CR99] similarly propose the use of external coordination messages to resolve problems relating to observability. Consider a pair of consecutive transitions x_j/y_j and x_{j+1}/y_{j+1} which may contain an undetectable 1-shift output fault. Their proposed method inserts an external coordination message exchange "$<-O_k, +O_h>$" relating to observability between x_j/y_j and x_{j+1}/y_{j+1} in the global test sequence. This results in the insertion of (In the following cases, tester k refers to the tester involved in the shift, and tester h refers to the tester sending the input to the IUT in x_{j+1}):

- Case 1: ($a_k \in y_{j+1}$) an external coordination message "$+O_h$" in the local test sequence for tester k just before receiving the output a_k from the IUT, sent by tester h. Tester h

will have a message "$-O_k$" inserted in its local test sequence just before sending the input x_{j+1} to the IUT. This message exchange has the effect of tester h telling tester k: "prepare to receive an output from the IUT."

- Case 2: ($a_k \in y_j$) an external coordination message "$+O_h$" in the local test sequence for tester k just after receiving the output a_k from the IUT, sent by tester h. Tester h will have a message "$-O_k$" inserted in its local test sequence just before sending the input x_{j+1} to the IUT. This message exchange has the effect of tester h telling tester k: "you should have received an output from the IUT by now."

To demonstrate, consider the digraph $G = (V, E)$ of the 2p-FSM $M5$ shown in Figure 13. The label of the transition tour $t1$, $t2$, $t3$, $t4$ forms a synchronizable global test sequence,

$\omega = a/<0, 1>, a/<0, 1>, b/<-, 1>, b/<0, ->,$

which corresponds to the following local test sequences for the Upper and Lower testers, respectively:

- $\omega_U = a/<0, ->, a/<0, ->, -/<-, ->, -/<0, ->$
- $\omega_L = -/<-, 1>, -/<-, 1>, b/<-, 1>, b/<-, ->$

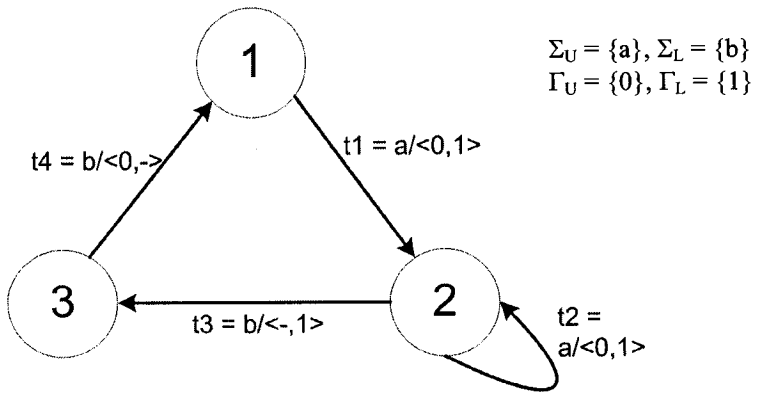


Figure 13. Digraph $G = (V, E)$ of 2p-FSM $M5$

A faulty implementation of $M5$ is shown in Figure 14, in which the output "0" intended for the Upper Tester in transition $t4$ has been shifted to transition $t3$.

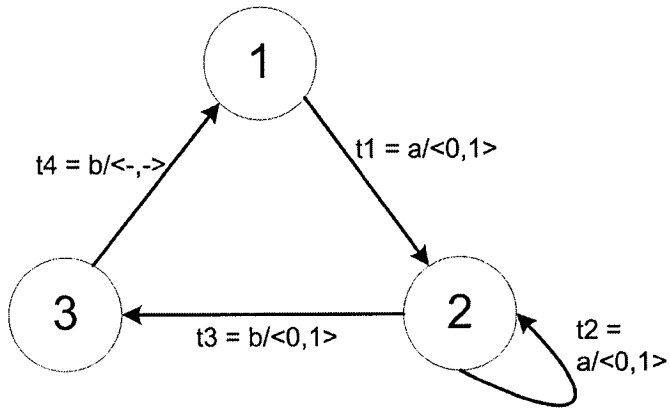


Figure 14. Faulty implementation of 2*p*-FSM *M5*

Note that 1-shift output fault in the implementation is not detectable by the application of the two local test sequences derived above; the Upper Tester receives the expected output from the IUT, but has no way of knowing that this output has been generated by the wrong transition.

To resolve this observability problem, the external coordination message exchange <-O_U, +O_L> relating to observability is inserted between b/<-, 1> and b/<0, -> in the global test sequence to detect the potential shift of the output "0". Thus, the above synchronizable global test sequence ω of $M5$ becomes:

a/<0, 1>, a/<0, 1>, b/<-, 1>, <-O_U, +O_L>, b/<0, ->,

resulting in the following local test sequences for the Upper and Lower testers, respectively:

ω_U = a/<0, ->, a/<0, ->, -/<-, ->, +O_L, -/<0, ->,

ω_L = -/<-, 1>, -/<-, 1>, b/<-, 1>, -O_U, b/<-, ->

In [HR00], Hierons proposes a method of augmenting the digraph G of an FSM M by adding edges representing test subsequences, then finding a minimal length tour of the

augmented digraph that contains each test subsequence. The method assumes a test architecture in which testers may communicate directly amongst themselves using external coordination messages. Similar to [CR99], Hierons proposes two types of observability-related coordination messages: Given two transitions t_{ij} and t_{jk}, a **post-transition framing message** is sent to the tester sending the input to the IUT in t_{jk} from each tester receiving an output from the IUT in t_{ij}, in order to preclude any possibility of a forward output shifting fault from t_{ij} to t_{jk}. If a tester is receiving an output in t_{ij} and sending an input in t_{jk}, no post-transition framing message for that tester is required. Given two transitions t_{ij} and t_{jk}, a **pre-transition framing message** is sent from the tester sending the input to the IUT in t_{jk} to each tester receiving an output from the IUT in t_{jk}, in order to preclude any possibility of a backward output-shifting fault from t_{jk} to t_{ij}.

In [CR99] and [HR00], external coordination message exchanges are inserted to resolve controllability and observability problems in the global test sequence. Whittier [WD01] improves upon this approach by considering the costs of external coordination message exchanges during the construction of the global test sequence. The method generates a test sequence that is minimal in terms of both the length of the sequence and the number of external coordination message exchanges. The fault model is limited to output faults and the method therefore generates only a transition tour for the given FSM.

The method proposed in [WD01] constructs three auxiliary graphs, the first two of which are used to address controllability problems. An RCPT over a subset of the edges of the third auxilary graph corresponds to a synchronizable global input-output sequence of M which is also free of potential undetectable 1-shift output faults.

The first auxiliary graph G' facilitates the generation of transition sequences that are free of controllability problems, and is constructed in three steps:

- Step 1. For each vertex $v_i \in V$
  - create a pair of vertices i^U and i^L in V', and
  - create a pair of dashed edges $(i^U, i^L; <-C_L, +C_U>)$ and $(i^L, i^U; <-C_U, +C_L>)$ in E', which indicate external coordination message exchanges relating to controllability.
- Step 2. For each edge $e_{jk} = (v_j, v_k; x/y) \in E$, create the following edges in E':
  - $(j^U, k^U; x/y)$, if $x \in \Sigma_U$, and $\nexists$ an $a_L \neq$ - in y,
  - $(j^U, k^U; x/y)$ and $(j^U, k^L; x/y)$, if $x \in \Sigma_U$ and $\exists$ an $a_L \neq$ - in y,
  - $(j^L, k^L; x/y)$, if $x \in \Sigma_L$, and $\nexists$ an $a_U \neq$ - in y,
  - $(j^L, k^L; x/y)$ and $(j^L, k^U; x/y)$, if $x \in \Sigma_L$ and $\exists$ an $a_U \neq$ - in y.
- Step 3. For each vertex $v \in V'$ where only dashed edges are arriving and leaving, remove from E' dashed edges arriving and leaving v and then remove v from V'.

In step 1 above, each vertex j^U (j^L) represents the starting state v_j of a transition with the input operation related to $U(L)$, and dashed edges represent an external coordination message exchange relating to controllability. The second step adds edges representing the transitions of M.

Consider the example 2p-FSM $M6$ from [WD01] shown in Figure 15. The corresponding digraph $G' = (V', E')$ is shown in Figure 16. Note that in any traversal of G', any two adjacent transitions of G will be covered without creating a synchronization problem. Note also that for any two states v_j and v_k, the shortest paths from j^U (or j^L) to

k^U (or k^L) can be used to generate the minimum-cost synchronizable sequences which transfer M from state v_j to v_k.

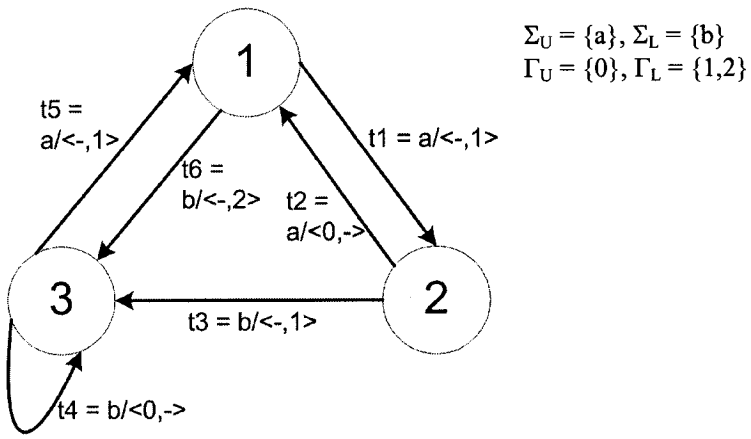


Figure 15. Digraph $G = (V, E)$ of $2p$-FSM $M6$

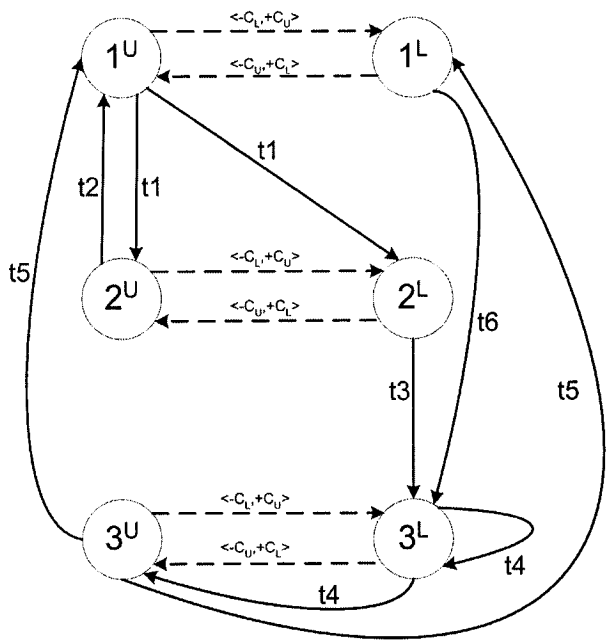


Figure 16. Digraph $G' = (V', E')$ of $2p$-FSM $M6$

One additional step generates a digraph $G'' = (V'', E'')$ from which a minimum-cost synchronizable global test sequence of M can be generated. This step proceeds as follows:

For each vertex v_i in V':

for each pair of edges (v_i, v_j^U; x/y) and (v_i, v_j^L; x/y):

- create a null vertex v_i^x in V'',
- create a solid bold edge (v_i, v_i^x, -/-) in E'',
- create two edges (v_i^x, v_j^U; x/y) and (v_i^x, v_j^L; x/y) in E'' and eliminate (v_i,v_j^U; x/y) and (v_i, v_j^L; x/y) from E''.

Any remaining solid edges leaving v_i are made bold.

The digraph $G''=(V'', E'')$ for 2p-FSM $M6$ is shown in Figure 17. Note that an RCPT of G'' over the set of bold edges results in a minimum-cost synchronizable global test sequence of M. To minimize the total number of external coordination message exchanges, each dashed edge can be given a sufficiently high cost so that the RCPT avoids these edges whenever possible. For 2p-FSM $M6$, such an RCPT yields a minimum-cost synchronizable global test sequence that does not use any external coordination message exchanges as the label of the path $P = t1, t3, t4, t5, t6, t4, t5, t1, t2$, for a total of nine input-output pairs.

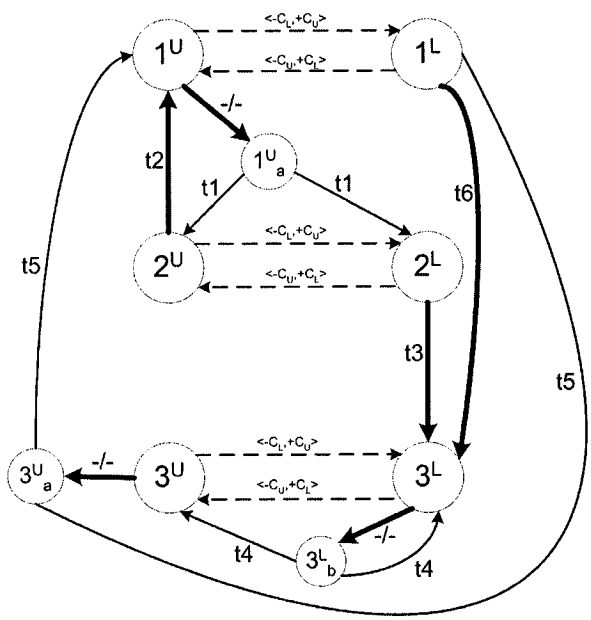


Figure 17. Digraph $G'' = (V'', E'')$ of 2p-FSM $M6$

The observability problem is addressed in [WD01] by first constructing from G the set of all transition pairs with a potential undetectable 1-shift output fault. This set, denoted T_O, and the digraph $G'' = (V'', E'')$ are used to form the digraph $G''' = (V''', E''')$, from which a global test sequence that is free of controllability and observability problems is generated.

$G''' = (V''', E''')$ is constructed in three steps:

- Step 1. For each triple $(t_{ij}, t_{jp}, <-O_k, +O_h>) \in T_O$, identify the vertex $v_j \in V''$ (i.e., either j^U or j^L) such that v_j is $tail(t_{ij})$. Add new vertex v_j* in V''' if one does not exist already.
- Step 2. For each triple $(t_{ij}, t_{jp}, <-O_k, +O_h>) \in T_O$, add a dashed edge from v_j* to v_j in to E''' with the label "t_{ij}, t_{jp}, $<-O_k$, $+O_h>$". If there already exists a dashed edge from v_j* to v_j labelled "t_{lm}, t_{mq}, $<-O_t$, $+O_u>$", then label "t_{ij}, t_{jp}, $<-O_k$, $+O_h>$" is added to this dashed edge as an alternate label.
- Step 3. For each solid edge t_{ij} in E'' whose label is not -/-, if t_{ij} leaves vertex v in V'' then it will leave the same vertex in V'''. If t_{ij} is contained in some triple $(t_{ij}, t_{jp}, <-O_k, +O_h>) \in T_O$ and arrives at a vertex v_j in V'', then t_{ij} will go to v_j* in V''', otherwise it will go to v_j in V'''.

Digraph $G''' = (V''', E''')$ for FSM $M6$ is shown in Figure 18. Note that any two consecutive transitions on a path in G that would be covered in the same order in a synchronizable traversal of G''' do so without any possibility of a potential undetectable 1-shift output fault. An RCPT of G''' over the set of bold edges yields a minimum-cost synchronizable global test sequence with no possibility of potential undetectable 1-shift output faults. To minimize external coordination message exchanges, each dashed edge

can again be given a sufficiently high cost so that an RCPT avoids these edges whenever possible. For FSM *M6* this yields a minimum-cost synchronizable global test sequence that has no potential undetectable 1-shift output faults as the label of:

$t1, t3, <\text{-}O_U, +O_L>, t4, <\text{-}O_L, +O_U>, t5, t6, <\text{-}O_U, +O_L>, t4, <\text{-}O_L, +O_U>, t5, t1, <\text{-}O_L, +O_U>, t2,$

which requires five external coordination message exchanges relating to observability.

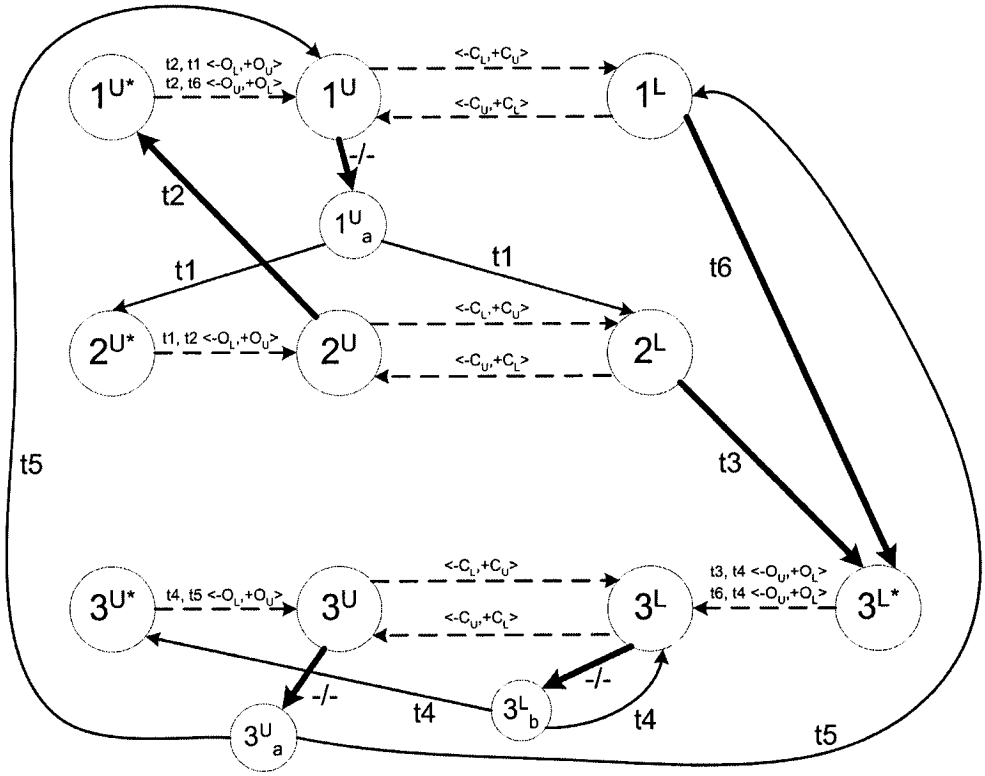


Figure 18. Digraph $G''' = (V''', E''')$ of 2*p*-FSM *M6*

Chapter Four

The Proposed Methods

4.1 Motivation

In the methods described in [CR99, LB94, YT98, WD01], the fault model assumed consists of only output faults; these methods therefore construct only a transition tour which verifies the output of each transition. These synchronizable global test sequences may not detect faults in an implementation that also has transfer faults. The checking sequence method presented in [HU02b] assumes a remote test architecture in which testers can only communicate via their interactions with the implementation under test. As a result, the method must make restrictive assumptions regarding the existence of synchronizable UIO sequences, and requires additional input sequences to address problems related to observability.

We propose two methods based on distinguishing sequences that will detect the presence of any output and/or transfer fault(s) in an implementation of an FSM. The first method assumes the presence of a reliable reset that returns the implementation to its initial state, while the second method does not. Both methods take into consideration both external coordination and input/output costs and consist of a set of transformation rules that construct modified digraphs from the specification of a given $2p$-FSM M, allowing for the construction of a synchronizable checking sequence that verifies every state and transition. This synchronizable checking sequence will ensure also that no 1-shift output fault remain undetected. Both methods attempt to reduce the total number of

external coordination message exchanges by taking their cost into consideration during the construction of the checking sequence.

Formally, the proposed methods are solutions of the following problem:

Consider a minimal and deterministic *np*-FSM $M = (S, \Sigma, \Gamma, \delta, \lambda, s_1)$ which is represented by a strongly connected digraph $G = (V, E)$ and has a distinguishing sequence D. Let $\Phi(M)$ be the set of all those implementations of M, each of which has the same sets of inputs and outputs as M, the same initial state as M, and at most $|S|$ states. Suppose that the test architecture to be used for testing implementations of M in $\Phi(M)$ consists of n testers and supports direct communication between testers through the use of a multicast channel and exchange of external coordination messages. Suppose also that the cost of executing an external coordination message is higher than that of a transition of M, and thus should be avoided when possible. Then, given M and $\Phi(M)$, construct a synchronizable checking sequence such that it

- distinguishes M from any faulty implementation of M in $\Phi(M)$,
- will not cause any controllability and observability problems when applied in a distributed test architecture, and
- is efficient in terms of both the number external coordination messages required and overall length of the checking sequence.

In the following sections we present the proposed methods as the solution of the above stated problem for an *np*-FSM where $n = 2$ and the ports are labelled U (Upper) and L (Lower). In Section 4.6, we extend the solutions for $n > 2$.

4.2 Proposed Method 1 – Reliable Reset

The method proposed in this section is composed of four phases:

- Phase 1 finds the set of all potential controllability problems and the set of all potential observability problems for the given FSM
- Phase 2 generates a test segment for each transition t_{ij}
- Phase 3 generates a preamble from s_1 to s_i for each state s_i
- Phase 4 constructs the state and transition covers and finds a synchronizable ordering of these subsequences, resulting in a synchronizable checking sequence with no potential undetectable 1-shift output faults.

In the first phase of the proposed method, the set of all controllability problems, T_C, and the set of all observability problems, T_O, are generated from the digraph $G = (V, E)$ of the given FSM M. The set T_C is constructed as follows:

for each vertex $v_j \in V$ *do*

for each edge e_{ij} *(say* t_{ij}*)* $= (v_i, v_j; x_j/y_j)$ *entering vertex* v_j *do*

for each edge e_{jk} *(say* t_{jk}*)*$= (v_j, v_k; x_{j+1}/y_{j+1})$ *leaving vertex* v_j *do*

if $x_j \in \Sigma_U$ *AND* $x_{j+1} \in \Sigma_L$ *AND* $a_L =$ *- in* y_j

then add $(t_{ij}, t_{jk}; <-C_L, +C_U>)$ *to* T_C

else if $x_j \in \Sigma_L$ *AND* $x_{j+1} \in \Sigma_U$ *AND* $a_U =$ *- in* y_j

then add $(t_{ij}, t_{jk}; <-C_U, +C_L>)$ *to* T_C

Each transition pair added to T_C forms a controllability problem as the sender of x_{j+1} is not the sender of x_j and does not receive an output in y_j. Given the label of a path P on G representing an input-output sequence, P can be made synchronizable as follows: For each consecutive transition pair t_{mn} t_{no} in P, if $(t_{mn}, t_{no}; <-C_{U(L)}, +C_{L(U)}>) \in T_C$ then insert

the external coordination message exchange $<-C_{U(L)}, +C_{L(U)}>$ relating to controllability between t_{mn} and t_{no} in the label of P.

Consider the FSM $M7$ shown in Figure 19. For readability, the transitions and their labels in the example FSMs discussed hereafter are numbered $t1$, $t2$, ... Applying the above procedure results in a set consisting of 1 non-synchronizable transition pair, i.e., $T_C = \{(t3, t10, <-C_U, +C_L>)\}$.

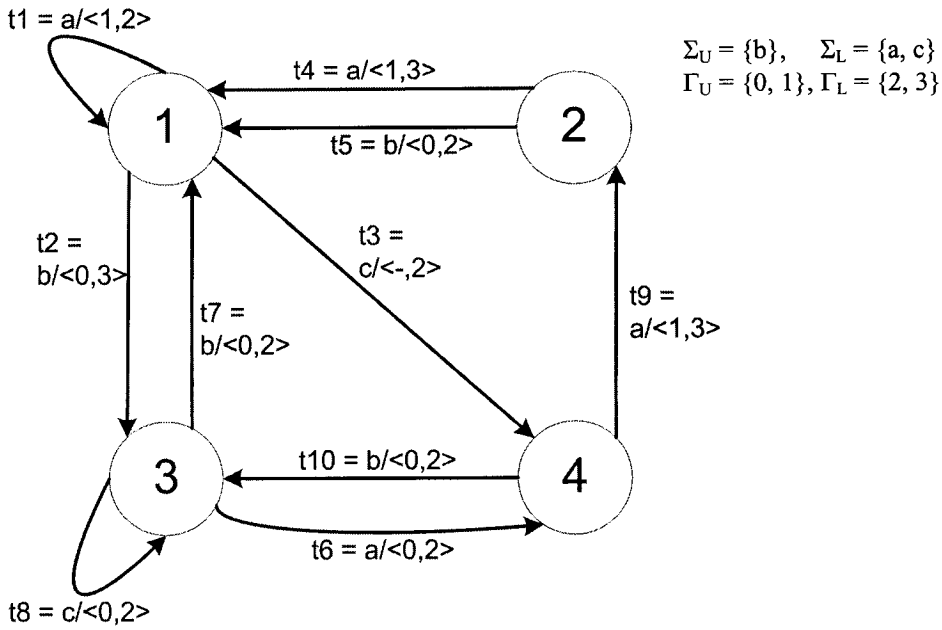


Figure 19. Digraph $G = (V, E)$ of 2p-FSM $M7$

The set T_O of all triples corresponding to transition pairs with a potential undetectable 1-shift output fault is generated next. The set T_O is constructed from $G = (V, E)$ as follows:

for each vertex $v_j \in V$ *do*

for each edge e_{ij} *(say* t_{ij}*)* $= (v_i, v_j; x_j/y_j)$ *entering vertex* v_j *do*

for each edge e_{jk} *(say* t_{jk}*)*$= (v_j, v_k; x_{j+1}/y_{j+1})$ *leaving vertex* v_j *do*

if for some output $a_{L(U)} \in \Gamma_{L(U)}$, $a_{L(U)}$ *is in* y_j *XOR* $a_{L(U)}$ *is in* y_{j+1}

AND $x_{j+1} \notin \Sigma_{L(U)}$ *AND* $(t_{ij}, t_{jk}; <-C_{U(L)}, +C_{L(U)}>) \notin T_C$

then add (t_{ij}, t_{jk}, <-$O_{L(U)}$, +$O_{U(L)}$>) *to* T_O,

where $U(L)$ is the tester sending the input x_{j+1} in t_{jk} and $L(U)$ is the tester involved in the shift.

The set T_O identifies *only the necessary subset* of all potential undetectable 1-shift output faults in G as defined in Section 2.3. Specifically, the *if-statement* in the algorithm limits T_O to only those transition pairs that form a synchronizable pair of transitions in G. Given the input and output alphabets shown in Figure 19, consider a pair of consecutive transitions t_{ij} = (s_i, s_j; a/<-,2>) and t_{jk} = (s_j, s_k; b/<0,->). This transition pair forms a potential undetectable forward shift fault of the output '2', i.e. L cannot determine whether '2' is output by a correctly implemented t_{ij}, or by faulty implementations of both t_{ij} and t_{jk}, i.e., t_{ij} = (s_i, s_j; a/<-,->) and t_{jk} = (s_j, s_k; b/<0,2>). However, note that any instance of t_{ij} followed by t_{jk} would not form a synchronizable pair of transitions and hence would require the insertion of an external coordination message exchange <-C_U,+C_L> relating to controllability. If we justifiably assume that L waits to receive the '2' from t_{ij} before sending the external coordination message $-C_U$ relating to controllability to U, the observability problem is resolved; i.e. if L waits and does not receive '2' before it sends $-C_U$, we conclude that the implementation of t_{ij} is faulty. A similar argument and intuitive treatment does not apply in any case of backward shifts if the output <-,-> is not allowed for any transition, other than the reset transitions.

Applying the above procedure to FSM $M7$ produces a set of 5 potential undetectable 1-shift output faults, i.e.:

T_O = {($t1$, $t3$, <-O_U, +O_L>), ($t3$, $t9$, <-O_U, +O_L>), ($t4$, $t3$, <-O_U, +O_L>),

($t5$, $t3$, <-O_U, +O_L>), ($t7$, $t3$, <-O_U, +O_L>)}

The second phase of the proposed method generates a test segment $test(t_{ij}) = t_{ij}@DS(s_j)$ for each transition t_{ij}, where $DS(s_i)$ is the transition sequence induced by D on G at v_i and $first(DS(s_i))$ denotes the first transition induced by D on G at v_i. Observability and controllability problems within $DS(s_i)$ and between t_{ij} and $first(DS(s_i))$ are resolved by inserting the corresponding external coordination message exchanges from T_O or T_C, respectively. Formally, this phase consists of two steps:

Step 1. For each state s_i, find the transition sequence $DS(s_i)$ induced by D on G at v_i.

For each pair of consecutive transitions t_{mn} t_{no} in $DS(s_i)$:

- If $(t_{mn}, t_{no}, <-O_{U(L)}, +O_{L(U)}>) \in T_O$ then insert the external coordination message exchange $<-O_{U(L)}, +O_{L(U)}>$ between t_{mn} and t_{no} in $label(DS(s_i))$ and remove $(t_{mn}, t_{no}, <-O_{U(L)}, +O_{L(U)}>)$ from T_O.
- If $(t_{mn}, t_{no}, <-C_{U(L)}, +C_{L(U)}>) \in T_C$ then insert the external coordination message exchange $<-C_{U(L)}, +C_{L(U)}>$ between t_{mn}, and t_{no} in $label(DS(s_i))$.

Step 2. For each transition $t_{ij} = (v_i, v_j; x_j/y_j)$:

- Construct $test(t_{ij}) = t_{ij}@DS(s_j)$.
- If there is a triple $(t_{mn}, t_{no}, <-O_{U(L)}, +O_{L(U)}>) \in T_O$ where $t_{mn} = t_{ij}$ and $t_{no} = first(DS(s_j))$ then insert the external coordination message exchange $<-O_{U(L)}, +O_{L(U)}>$ between t_{ij} and $DS(s_j)$ in $label(test(t_{ij}))$ and remove $(t_{mn}, t_{no}, <-O_{U(L)}, +O_{L(U)}>)$ from T_O.
- If there is a triple $(t_{mn}, t_{no}, <-C_{U(L)}, +C_{L(U)}>) \in T_C$ where $t_{mn} = t_{ij}$ and $t_{no} = first(DS(s_j))$ then insert the external coordination message exchange $<-C_{U(L)}, +C_{L(U)}>$ between t_{ij} and $DS(s_j)$ in $label(test(t_{ij}))$.

The distinguishing sequence D for FSM $M7$is ab. The transition sequences induced by this D at each state are shown in Table 3, and the test segments generated in Step 2 are shown in Table 4. Note that Step 2 removes triple ($t3$, $t9$, $<-O_U, +O_L>$) from T_O as each potential 1-shift output fault needs only be handled once. Removing triples ensures that the remaining phases of the proposed method do not unnecessarily avoid transition pairs whose potential 1-shift output fault is already handled in some $test(t_{ij})$.

Table 3. *label*($DS(s_i)$) for 2*p*-FSM *M7*

State s_i	$label(DS(s_i))$
s_1	$t1$ $t2$
s_2	$t4$ $t2$
s_3	$t6$ $t10$
s_4	$t9$ $t5$

Table 4. Test segments for 2*p*-FSM *M7*

Transition t_{ij}	$label(test(t_{ij}))$
$t1$	$t1$ $t1$ $t2$
$t2$	$t2$ $t6$ $t10$
$t3$	$t3$ $<-O_U, +O_L>$ $t9$ $t5$
$t4$	$t4$ $t1$ $t2$
$t5$	$t5$ $t1$ $t2$
$t6$	$t6$ $t9$ $t5$
$t7$	$t7$ $t1$ $t2$
$t8$	$t8$ $t6$ $t10$
$t9$	$t9$ $t4$ $t2$
$t10$	$t10$ $t6$ $t10$

The third phase of the proposed method generates a preamble for each state s_i, which is a transition sequence that transfers M from the initial state s_1 to state s_i. In choosing a preamble for state s_i, denoted $preamble(s_i)$, three goals must be considered:

1) Minimize observability and controllability problems in $preamble(s_i)$
2) Minimizing observability and controllability problems between the last transition of $preamble(s_i)$ and transitions starting at s_i.

3) Minimize the length of *preamble*(s_i)

These goals may conflict; the preamble for s_i that requires the fewest external coordination message exchanges may end with a transition which causes significant problems when followed by the transitions starting at s_i. As our goal is to minimize the number of external coordination message exchanges introduced by the use of *preamble*(s_i), goals 1 and 2 are given precedence and both these goals are considered in choosing *preamble*(s_i). This is accomplished by first calculating a cost for a transition t_{ij} based on the number of controllability and observability problems that will be introduced if a sequence ending with transition t_{ij} is chosen as *preamble*(s_j). Using these costs, preambles for each state $s_j \neq s_1$ are found by first constructing a graph $G' = (V', E')$. Edges in E' are assigned a cost based on the controllability and observability problems caused by transition pairs represented by adjacent edges. For each state s_j, a graph G_j is created from G'. An RCPT over a selected edge in G_j selects the preamble for state s_j that causes the fewest controllability and observability problems in the resulting state cover and transition cover sequences. Formally, this phase proceeds as follows:

Step 1. The sum of the external coordination message exchange costs for each transition t_{ij} (that may be the last transition in a preamble for state s_j) followed by any of its adjacent transitions is calculated as follows:

for each vertex $v_j \in V$, $v_j \neq v_1$, *do*

for each edge e_{ij} (*say* t_{ij}) = (v_i, v_j; x_j/y_j) *entering vertex* v_j *where* $v_i \neq v_j$ *do*

let $sum\_cost(t_{ij}) = 0$

for each edge e_{jk} (*say* t_{jk})= (v_j, v_k; x_{j+1}/y_{j+1}) *leaving vertex* v_j *do*

if (t_{ij}, t_{jk}, $<-C_{U(L)}, +C_{L(U)}>$) $\in T_C$

if t_{jk} = *first*($DS(s_j)$)

then $sum\_cost(t_{ij}) = sum\_cost(t_{ij}) + 2w$ (*as the pair* t_{ij} t_{jk} *will occur twice, once in verifying* s_j *and once to verify* t_{jk})

else $sum\_cost(t_{ij}) = sum\_cost(t_{ij}) + w$

if $(t_{ij}, t_{jk}, <-O_{U(L)}, +O_{L(U)}>) \in T_O$

then $sum\_cost(t_{ij}) = sum\_cost(t_{ij}) + w$

Step 2. Construct the graph $G' = (V', E')$ from $G = (V, E)$ by the following steps:

create a vertex v_1 *in* V'

for each edge e_{1k} (*say* t_{1k}) = $(v_1, v_k; x_k/y_k)$ *leaving vertex* $v_1 \in V$ *where* $v_k \neq v_1$ *do*

create a vertex labelled "v_1-t_{1k}-v_k*" in* V'

add an edge from v_1 *to "*v_1-t_{1k}-v_k*" labelled* t_{1k}, *i.e.,* $(v_1, v_1\text{-}t_{1k}\text{-}v_k; t_{1k})$

let $cost(t_{1k}) = 1$

for each vertex $v_j \in V$, $v_j \neq v_1$

for each edge e_{ij} (*say* t_{ij}) = $(v_i, v_j; x_j/y_j)$ *entering vertex* v_j *in* V, $head(e_{ij}) \neq v_j$:

for each edge e_{jk} (*say* t_{jk}) = $(v_j, v_k; x_k/y_k)$ *leaving vertex* $v_j \in V$, $tail(e_{jk}) \neq v_j$, $tail(e_{jk}) \neq v_1$:

create a vertex labelled "v_i-t_{ij}-v_j*" in* V' *if one does not exist already*

create a vertex labelled "v_j-t_{jk}-v_k*" in* V' *if one does not exist already*

add an edge $e'_{jk} = (v_i\text{-}t_{ij}\text{-}v_j, v_j\text{-}t_{jk}\text{-}v_k; t_{jk})$ *in* E'

if $\exists$ *a triple* $(t_{ij}, t_{jk}, <-O_{U(L)}, +O_{L(U)}>) \in T_O$ *then* $cost(e'_{jk}) = 1 + w$

if $\exists$ *a triple* $(t_{ij}, t_{jk}, <-C_{U(L)}, +C_{L(U)}>) \in T_C$

then $cost(e'_{jk}) = 1 + w^*(m + 1)$

else $cost(e'_{jk}) = 1$

Step 3. For each state $s_j \neq s_1$, create a graph $G_j = (V_j, E_j)$ from $G' = (V', E')$ by the following:

Initially, $G_j = G'$

create a vertex v' *in* V_j *and add an edge* $Z = (v', v_1; \text{"}Z\text{"})$ *to* E_j

for each vertex labelled "v_i-t_{ij}-v_j*" in* V'

add a dashed edge $e'_{ij} = (v_i\text{-}t_{ij}\text{-}v_j, v'; -)$ *to* E_j

let $cost(e'_{ij}) = sum\_cost(t_{ij})$ *(as calculated in step 1)*

Step 4. For each state $s_j \neq s_1$: Find an RCPT P of G_j, starting at v_1, over the single edge Z.

Let $preamble(s_j) = label(P')$ which is a transition sequence where P' is the subpath of P from v_1 to v_i-t_{ij}-v_j.

For each pair of consecutive transitions t_{mn} t_{no} in $preamble(s_j)$:

- if $(t_{mn}, t_{no}, <\text{-}C_{U(L)}, +C_{L(U)}>) \in T_C$ then insert the external coordination message exchange $<\text{-}C_{U(L)}, +C_{L(U)}>$ between t_{mn} and t_{no} in $label(preamble(s_j))$.
- if $(t_{mn}, t_{no}, <\text{-}O_{U(L)}, +O_{L(U)}>) \in T_O$ then insert the external coordination message exchange $<\text{-}O_{U(L)}, +O_{L(U)}>$ between t_{mn} and t_{no} in $label(preamble(s_j))$.

In Step 2, the cost of each edge leaving v_1 is 1. The cost of every remaining edge e'_{jk} in E' depends on whether the transition pair $(v_i, v_j; x_j/y_j)$ $(v_j, v_k; x_k/y_k)$ appears in T_O or T_C. The variable w represents a high cost to be associated with external coordination message exchanges. The cost of $w*(m + 1)$ for a controllability problem is based on the following: The purpose of G' is to aid in choosing preambles which minimize the number of external coordination message exchanges required in the resulting sequences forming the state

cover and transition cover. Each preamble will be used once to form a state cover sequence for state s_i, and m times to verify each of the m transitions starting at s_i. Therefore if a transition pair in a preamble contains a controllability problem, the resulting number of external coordination message exchanges introduced is $(m + 1)$, so the cost assigned in E' is $w^*(m + 1)$. When constructing the graph G_j, the outdegree of node v_j in G is then substituted for m. In contrast, observability problems result in a cost of w as each potential 1-shift output fault need only be checked once. If the pair $(v_i, v_j; x_j/y_j)$ $(v_j, v_k; x_k/y_k)$ does not appear in a triple in T_O or T_C, the cost of edge e'_{jk} is 1.

For FSM *M7*, Step 1 calculates the costs shown in Table 5. Note that the transitions not included are those entering state s_1, as s_1 does not require a preamble, and those that start and end at the same state.

Table 5. *sum_cost*(t_{ij}) for transition t_{ij}

Transition t_{ij}	*sum_cost*(t_{ij})
*t*2	0
*t*6	0
*t*10	0
*t*3	*w*
*t*9	0

Applying Step 2 to FSM *M7* generates the graph $G' = (V', E')$ shown in Figure 20, with the cost of each edge shown in parentheses. Based on graphs G_2, G_3, and G_4, preambles selected for states s_2, s_3, s_4, of FSM *M7* are *t*3 *t*9, *t*2, and *t*2 *t*6, respectively. As an example, the Digraph $G_3 = (V_3, E_3)$ created for s_3 is shown in Figure 21.

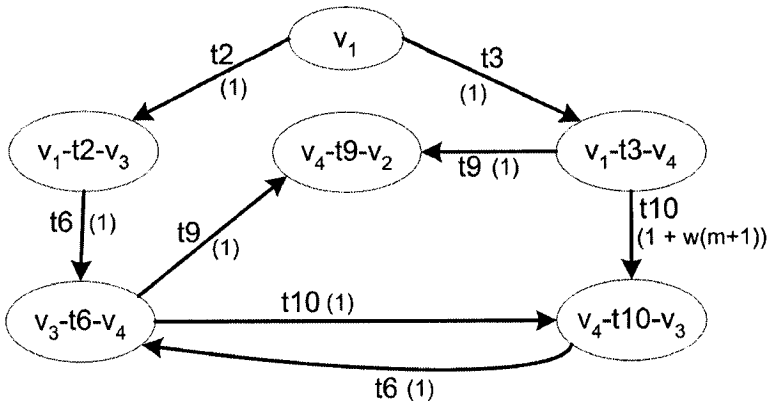


Figure 20. Digraph $G' = (V', E')$ of 2p-FSM $M7$

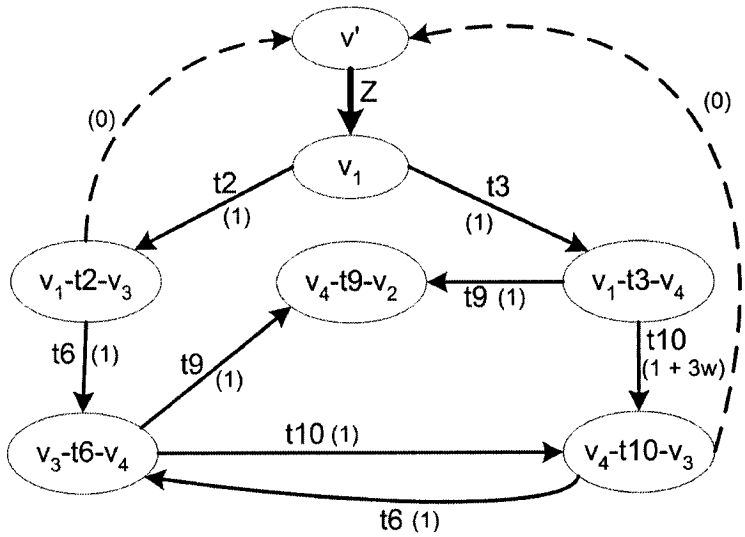


Figure 21. Digraph $G_3 = (V_3, E_3)$ of 2p-FSM $M7$

The final phase of the proposed method first generates sequences that form the state cover and transition cover, using the preambles found in the previous phase. Following prefix elimination, an RCPP on an auxiliary graph G'' yields a synchronizable ordering of the remaining sequences. The input portion of the transition sequence represented by the label of this path is a synchronizable checking sequence with no potential undetectable 1-shift output faults for FSM M. This phase proceeds as follows:

Step 1. For each state s_j:

- Let $state\_cover(s_j) = preamble(s_j)@DS(s_j)$

- If $\exists$ a triple $(t_{mn}, t_{no}, <\text{-}C_{U(L)}, +C_{L(U)}>) \in T_C$ where t_{mn} = *last*(*preamble*(s_j)) and t_{no} = *first*(*DS*(s_j)) then insert the external coordination message exchange $<\text{-}C_{U(L)}, +C_{L(U)}>$ between t_{mn} and t_{no} in *label*(*state_cover*(s_j)).
- If $\exists$ a triple $(t_{mn}, t_{no}, <\text{-}O_{U(L)}, +O_{L(U)}>) \in T_O$ where t_{mn} = *last*(*preamble*(s_j)) and t_{no} = *first*(*DS*(s_j)) then insert the external coordination message exchange $<\text{-}O_{U(L)}, +O_{L(U)}>$ between t_{mn} and t_{no} in *label*(*state_cover*(s_j)).

Step 2. For each transition $t_{jk} = (s_j, s_k; x/y)$:

- Let *trans_cover*(t_{jk}) = *preamble*(s_j)@*test*(t_{jk})
- If $\exists$ a triple $(t_{mn}, t_{no}, <\text{-}C_{U(L)}, +C_{L(U)}>) \in T_C$ where t_{mn} = *last*(*preamble*(s_j)) and t_{no} = t_{jk} then insert the external coordination message exchange $<\text{-}C_{U(L)}, +C_{L(U)}>$ between t_{mn} and t_{no} in *label*(*trans_cover*(t_{jk})).
- If $\exists$ a triple $(t_{mn}, t_{no}, <\text{-}O_{U(L)}, +O_{L(U)}>) \in T_O$ where t_{mn} = *last*(*preamble*(s_j)) and t_{no} = t_{jk} then insert the external coordination message exchange $<\text{-}O_{U(L)}, +O_{L(U)}>$ between t_{mn} and t_{no} in *label*(*trans_cover*(t_{jk})).

Step 3. Let C be the set of all transition sequences in the state and transition covers. For every transition sequence $c_p \in C$, if c_p is a prefix of some c_q, $c_q \in C$, $c_p \neq c_q$, then for any external coordination message exchange $<\text{-}O_{U(L)}, +O_{L(U)}>$ relating to observability between transitions t_{mn} and t_{no} in *label*(c_p), insert this observability message between t_{mn} and t_{no} in *label*(c_q). Then eliminate c_p.

Step 4. For the f sequences $c_1, \ldots, c_f$ remaining in C, let $l_i = label(c_i)$, $1 \leq i \leq f$. For every subsequence $label(t_{mn})$ $<-O_{U(L)}, +O_{L(U)}>$ $label(t_{no})$ in any label l_i, remove $<-O_{U(L)}, +O_{L(U)}>$ from any subsequent occurrence of this subsequence in any label l_j.

In the following steps, h is the tester sending the *last* input of D, and h' is the other tester.

Step 5. Create four vertices in V'' labelled 1^U, 1^L, T^B, and T^h.

Step 6. For each sequence $c_i \in C$, a solid edge is added to E'' as follows:

- $(1^U, T^h; l_i)$, if sender of x in $first(c_i)$ is U, and $last(c_i)$ sends output only to h,
- $(1^U, T^B; l_i)$, if sender of x in $first(c_i)$ is U, and $last(c_i)$ sends output to h',
- $(1^L, T^h; l_i)$, if sender of x in $first(c_i)$ is L, and $last(c_i)$ sends output only to h,
- $(1^L, T^B; l_i)$, if sender of x in $first(c_i)$ is L, and $last(c_i)$ sends output to h'.

Step 7. Add the following dashed edges to E'' representing reset transitions:

- $(T^B, 1^U;$ "RfU/-") and $(T^B, 1^L;$ "RfL/-")
- $(T^h, 1^h;$ "Rfh/-")

Step 8. Add a dashed edge $(1^h, 1^{h'}; <-C_{h'}, +C_h>)$ to E''.

After this step is complete, the resulting digraph will be known as $G'' = (V'', E'')$.

Step 9. Beginning at $1^{h'}$, find a rural Chinese postman path (RCPP) over the solid edges in E''. The input portion of the label of this path represents a checking sequence for FSM M.

Step 10. Eliminate any external coordination message exchanges in the resulting checking sequence that relate to potential 1-shift output faults that can be rendered detectable by some subsequence in the checking sequence, as in [LB94].

The first two steps of this phase generate the state and transition covers, respectively. In the prefix elimination in Step 3, a transition sequence c_p is eliminated if it is a prefix

(without considering external coordination message exchanges) of some other sequence c_q. However, if $label(c_p)$ contains external coordination messages relating to observability, these messages are first copied into $label(c_q)$ to ensure that all potential undetectable output shift faults remain detectable. Redundant external coordination message exchanges relating to observability are removed in Step 4.

In Step 5, vertices 1^L and 1^U are created as all sequences in C begin at the initial state with input from U or L. Step 6 adds solid edges representing sequences $c_1, \ldots, c_f$. Edges that terminate at T^B can be followed by a reset input from either tester. If the last transition of c_i sends output only to h, the edge terminates at T^h and may only be followed by a reset input from h. In Step 7, dashed edges representing these reset inputs from U and L are added to E'' as 'rfU/-' and 'rfL/-' respectively. The dashed edge (1^h, $1^{h'}$; $<-C_{h'}$, $+C_h>$) added in Step 8 represents an external coordination message exchange relating to controllability that may be used during the construction of the rural Chinese postman path over the set of solid edges in E''.

Applying steps 1 and 2 to the example FSM $M7$ yield the state and transition covers shown in Table 6 and Table 7, respectively.

Table 6. State cover for 2p-FSM $M7$

State s_i	$label(state\_cover(s_i))$
s_1	$t1$ $t2$
s_2	$t3$ $t9$ $t4$ $t2$
s_3	$t2$ $t6$ $t10$
s_4	$t2$ $t6$ $t9$ $t5$

Table 7. Transition cover for 2*p*-FSM *M*7

Transition t_{ij}	*label*(*trans_cover*(t_{ij}))
*t*1	*t*1 *t*1 *t*2
*t*2	*t*2 *t*6 *t*10
*t*3	*t*3 <-O_U, +O_L> *t*9 *t*5
*t*4	*t*3 *t*9 *t*4 *t*1 *t*2
*t*5	*t*3 *t*9 *t*5 *t*1 *t*2
*t*6	*t*2 *t*6 *t*9 *t*5
*t*7	*t*2 *t*7 *t*1 *t*2
*t*8	*t*2 *t*8 *t*6 *t*10
*t*9	*t*2 *t*6 *t*9 *t*4 *t*2
*t*10	*t*2 *t*6 *t*10 *t*6 *t*10

In Step 3, 4 sequences are eliminated as shown in Table 8. The set of 10 remaining sequences is shown in Table 9; note that the observability message in label of the eliminated sequence *trans_cover*(*t*3) has been copied into l_5 in Step 4.

Table 8. Sequences eliminated in Step 3 for 2*p*-FSM *M*7

Sequence Eliminated	Prefix of
state_cover(s_3)	*trans_cover*(*t*10)
state_cover(s_4)	*trans_cover*(*t*6)
trans_cover(*t*2)	*trans_cover*(*t*10)
trans_cover(*t*3)	*trans_cover*(*t*5)

Table 9. Checking sequence subsequences for 2*p*-FSM *M*7

Label	Transition Sequence	Verifies
l_1	*t*1 *t*2	s_1
l_2	*t*3 *t*9 *t*4 *t*2	s_2
l_3	*t*1 *t*1 *t*2	*t*1
l_4	*t*3 *t*9 *t*4 *t*1 *t*2	*t*4
l_5	*t*3 <-O_U, +O_L> *t*9 *t*5 *t*1 *t*2	*t*3, *t*5
l_6	*t*2 *t*6 *t*9 *t*5	s_4, *t*6
l_7	*t*2 *t*7 *t*1 *t*2	*t*7
l_8	*t*2 *t*8 *t*6 *t*10	*t*8
l_9	*t*2 *t*6 *t*9 *t*4 *t*2	*t*9
l_{10}	*t*2 *t*6 *t*10 *t*6 *t*10	s_3, *t*2, *t*10

Figure 22 shows the digraph G″ for *M*7 obtained by Step 5 to Step 8. A rural Chinese postman path over the solid edges obtained in Step 9 yields a synchronizable

checking sequence, with no potential undetectable 1-shift output faults, represented on G″ by the input portion of the path represented by the label sequence:

rfL/- l_1 rfL/- l_2 rfL/- l_3 rfL/- l_4 rfL/- l_5 rfU/- l_6 rfU/- l_7 rfU/- l_8 rfU/- l_9 rfU/- l_{10}

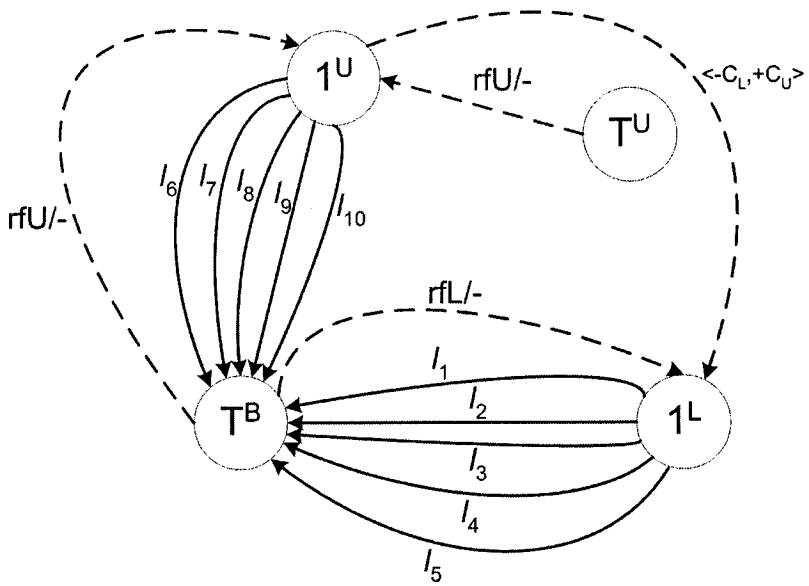


Figure 22. Digraph $G'' = (V'', E'')$ of 2p-FSM $M7$

The input portion of the path represented by this label sequence on G″ corresponds to a checking sequence composed of 10 reset inputs, 41 non-reset inputs, 1 external coordination message exchange relating to observability, and no external coordination message exchanges relating to controllability.

In Step 10, the subsequence $t6$ $t9$ $t5$ in the checking sequence above is found to be sufficient to detect the potential backward shift of '1' in $t3$ $t9$, thus rendering the external coordination message exchange $<-O_U, +O_L>$ between $t3$ and $t9$ unnecessary. As a result, the checking sequence generated by our method for FSM $M7$ requires no external coordination message exchanges relating to controllability and observability.

A proof that checking sequences generated by this method are free of controllability problems and undetectable 1-shift output faults is given in Section 4.4.

4.3 Proposed Method 2 – α'-sequences

The method proposed in this section is composed of five phases:

- Phase 1 finds the set of all potential controllability problems and the set of all potential observability problems for the given FSM
- Phase 2 finds the transition sequence induced by D at each state s_i, and resolves any potential controllability and observability problems which may occur between each transition t_{ij} and $first(DS(s_j))$
- Phase 3 constructs an auxiliary graph G' which is then used to generate a T-sequence T_i for each state s_i of the given FSM
- Phase 4 constructs α'-sequences from the T-sequences generated in Phase 3
- Phase 5 constructs a digraph $G'' = (V'', E'')$ and its minimal symmetric augmentation $G''' = (V''', E''')$. An Euler tour of G''' represents a synchronizable checking sequence with no potential undetectable 1-shift output faults.

The first phase of this proposed method is identical to that of the method presented in Section 4.2. The set of all controllability problems, T_C, and the set of all observability problems, T_O, are generated from the digraph $G = (V, E)$ of the given FSM M. Formally, this phase proceeds as follows:

Step 1. The set T_C is constructed from $G = (V, E)$ as follows:

for each vertex $v_j \in V$ *do*

for each edge e_{ij} *(say* t_{ij}*)* $= (v_i, v_j; x_j/y_j)$ *entering vertex* v_j *do*

for each edge e_{jk} *(say* t_{jk}*)*$= (v_j, v_k; x_{j+1}/y_{j+1})$ *leaving vertex* v_j *do*

if $x_j \in \Sigma_U$ *AND* $x_{j+1} \in \Sigma_L$ *AND* $a_L =$ *- in* y_j

then add $(t_{ij}, t_{jk}; <-C_L, +C_U>)$ *to* T_C

else if $x_j \in \Sigma_L$ *AND* $x_{j+1} \in \Sigma_U$ *AND* a_U = - *in* y_j

then add $(t_{ij}, t_{jk}; <-C_U, +C_L>)$ *to* T_C

Step 2. The set T_O is constructed from $G = (V, E)$ as follows:

for each vertex $v_j \in V$ *do*

for each edge e_{ij} *(say* t_{ij}*)* = $(v_i, v_j; x_j/y_j)$ *entering vertex* v_j *do*

for each edge e_{jk} *(say* t_{jk}*)*= $(v_j, v_k; x_{j+1}/y_{j+1})$ *leaving vertex* v_j *do*

if for some output $a_{L(U)} \in \Gamma_{L(U)}$, $a_{L(U)}$ *is in* y_j *XOR* $a_{L(U)}$ *is in* y_{j+1}

AND $x_{j+1} \notin \Sigma_{L(U)}$ *AND* $(t_{ij}, t_{jk}; <-C_{U(L)}, +C_{L(U)}>) \notin T_C$

then add $(t_{ij}, t_{jk}, <-O_{L(U)}, +O_{U(L)}>)$ *to* T_O,

where $U(L)$ is the tester sending the input x_{j+1} in t_{jk} and $L(U)$ is the tester involved in the shift.

Consider the example FSM $M8$ shown in Figure 23. The set T_C contains 2 triples while T_O contains 6 triples, i.e.:

T_C ={($t8$, $t7$; <-C_U, +C_L>), ($t9$, $t5$; <-C_U, +C_L>)}

T_O = {($t1$, $t8$; <-O_U, +O_L>), ($t3$, $t9$; <-O_U, +O_L>), ($t7$, $t9$; <-O_U, +O_L>), ($t8$, $t6$; <-O_U, +O_L>), ($t9$, $t4$; <-O_U, +O_L>), ($t10$, $t8$; <-O_U, +O_L>)}

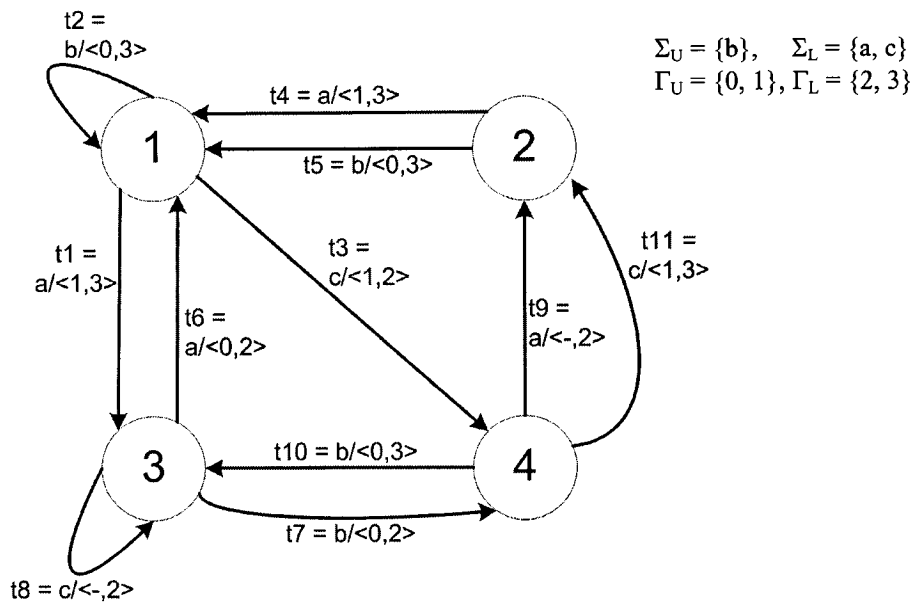


Figure 23. Digraph $G = (V, E)$ of 2p-FSM $M8$

The second phase finds the transition sequence induced by D at each state s_i, denoted $DS(s_i)$. Observability and controllability problems within $DS(s_i)$ are resolved by inserting the corresponding external coordination message exchanges from T_O or T_C, respectively. Problems between each transition t_{ij} and $first(DS(s_j))$ are similarly resolved. A cost is associated with each $DS(s_i)$ based on the number of controllability problems it contains; any $DS(s_i)$ with a high cost can then be avoided if possible in the subsequences created in subsequent phases, thus reducing the total number of external coordination message exchanges in the resulting checking sequence. Removing triples from T_O ensures that the remaining phases of the proposed method do not unnecessarily avoid transition pairs whose potential 1-shift output fault is already handled. Formally, this phase consists of two steps:

Step 1. For each state s_i, find the transition sequence $DS(s_i)$ induced by D on G at v_i.

Initially, $cost(DS(s_i)) = 0$.

For each pair of consecutive transitions t_{mn} t_{no} in $DS(s_i)$:

- If $(t_{mn}, t_{no}, <-C_{U(L)}, +C_{L(U)}>) \in T_C$ then insert the external coordination message exchange $<-C_{U(L)}, +C_{L(U)}>$ between t_{mn} and t_{no} in $label(DS(s_i))$ and let $cost(DS(s_i)) = cost(DS(s_i)) + w$.
- If $(t_{mn}, t_{no}, <-O_{U(L)}, +O_{L(U)}>) \in T_O$ then insert the external coordination message exchange $<-O_{U(L)}, +O_{L(U)}>$ between t_{mn} and t_{no} in $label(DS(s_i))$ and remove $(t_{mn}, t_{no}, <-O_{U(L)}, +O_{L(U)}>)$ from T_O.

Step 2. For each edge e_{ij} (*say* t_{ij}) = $(v_i, v_j; x_j/y_j)$ in E

If $(t_{ij}, first(DS(s_j)), <-C_{U(L)}, +C_{L(U)}>) \in T_C$

then $pre\_test(t_{ij}) = label(t_{ij}) <-C_{U(L)}, +C_{L(U)}>$

else if $(t_{ij}, first(DS(s_j)), <-O_{U(L)}, +O_{L(U)}>) \in T_O$

then $pre\_test(t_{ij}) = label(t_{ij}) <-O_{U(L)}, +O_{L(U)}>$

remove $(t_{ij}, first(DS(s_j)), <-O_{U(L)}, +O_{L(U)}>)$ from T_O

else $pre\_test(t_{ij}) = label(t_{ij})$

Note that in Step 2, the label $pre\_test(t_{ij})$ denotes only the transition t_{ij} and any external coordination message exchange required between t_{ij} and $first(DS(s_j))$, not a complete test segment $t_{ij}@DS(s_j)$. Test segments for each transition will be formed in a later phase based on the same graph construction technique as in [HU02a].

The distinguishing sequence D for 2p-FSM $M8$ is ab. The transition sequence $DS(s_i)$ and its cost for each state s_i is shown in Table 10, and $pre\_test(t_{ij})$ for each transition t_{ij} is shown in Table 11. Note that the triples $\{(t3, t9; <-O_U, +O_L>), (t7, t9; <-O_U, +O_L>), (t8, t6; <-O_U, +O_L>), (t9, t4; <-O_U, +O_L>)\}$ are removed from T_O in Step 2, resulting in $T_O = \{(t1, t8; <-O_U, +O_L>), (t10, t8; <-O_U, +O_L>)\}$.

Table 10. ***label*(*DS*(s_i)) and *cost*(*DS*(s_i)) for 2*p*-FSM *M*8**

State s_i	*label*(*DS*(s_i))	*cost*(*DS*(s_i))	*Outputs*	$\delta(s_i, D)$
s_1	$t1\ t7$	0	<1,3><0,2>	s_4
s_2	$t4\ t2$	0	<1,3><0,3>	s_1
s_3	$t6\ t2$	0	<0,2><0,3>	s_1
s_4	$t9$ <-C_U, +C_L> $t5$	w	<-,2><0,3>	s_1

Table 11. ***pre_test*(t_{ij}) for 2*p*-FSM *M*8**

Transition t_{ij}	*pre_test*(t_{ij})
$t1$	$t1$
$t2$	$t2$
$t3$	$t3$ <-O_U, +O_L>
$t4$	$t4$
$t5$	$t5$
$t6$	$t6$
$t7$	$t7$ <-O_U, +O_L>
$t8$	$t8$ <-O_U, +O_L>
$t9$	$t9$ <-O_U, +O_L>
$t10$	$t10$
$t11$	$t11$

In Phase 3, a digraph G' is constructed to aid in generating T-sequences and α'-sequences. A T-sequence T_j begins with *label*(*DS*(s_j)) and thus may be used to verify the end state of any transition t_{ij}. An α'-sequence, constructed from T-sequences, can similarly be used to verify the end state of a transition, and also recognizes a subset of the states of the FSM. G' is constructed such that problems relating to controllability and observability in T-sequences generated using G', and resulting α'-sequences, are minimized. This is achieved by assigning a cost to edges in E' based on the controllability and observability problems caused by transition pairs represented by adjacent edges. This phase consists of 2 steps:

Step 1. Construct the digraph $G' = (V', E')$ from $G = (V, E)$ as follows:

for each vertex $v_j \in V$

for each edge e_{ij} *(say* t_{ij}*)* $= (v_i, v_j; x_j/y_j)$ *entering vertex* v_j *in* V, $head(e_{ij}) \neq v_j$:

for each edge e_{jk} *(say* t_{jk}*)* $= (v_j, v_k; x_k/y_k)$ *leaving vertex* $v_j \in V$, $tail(e_{jk}) \neq v_j$:

create a vertex labelled "v_i-t_{ij}-v_j*" in* V' *if one does not exist already*

create a vertex labelled "v_j-t_{jk}-v_k*" in* V' *if one does not exist already*

add an edge $e'_{jk} = (v_i\text{-}t_{ij}\text{-}v_j, v_j\text{-}t_{jk}\text{-}v_k; t_{jk})$ *in* E'

if $\exists$ *a triple* $(t_{ij}, t_{jk}, <\text{-}O_{U(L)}, +O_{L(U)}>) \in T_O$ *OR a triple* $(t_{ij}, t_{jk}, <C_{U(L)}, +C_{L(U)}>) \in T_C$ *then* $cost(e'_{jk}) = 1 + w$

else $cost(e'_{jk}) = 1$

create a vertex labeled Z in V'

for each vertex $(v_i\text{-}t_{ij}\text{-}v_j)$ *in* V'

create a dashed edge $z_{ij} = ((v_i\text{-}t_{ij}\text{-}v_j), Z; <\text{-},\text{->})$

if $(t_{ij}, first(DS(s_j)), <\text{-}C_{U(L)}, +C_{L(U)}>) \in T_C$ *then* $cost(z_{ij}) = cost(DS(s_j)) + w$

else $cost(z_{ij}) = cost(DS(s_j))$

Step 2. For each state s_i, the T-sequence T_i is selected as follows:

- On G', find the min-cost path P_i from $(head(last(DS(s_i)))\text{-}last(DS(s_i))\text{-}tail(last(DS(s_i))))$ to Z.
- If there is more than one such min-cost path, P_i is chosen as the path for which the last transition in the path occurs least often as transition t_{jk} in elements $(t_{jk}, t_{kl}, <\text{-}O_{U(L)}, +O_{L(U)}>) \in T_O$ and $(t_{jk}, t_{kl}, <\text{-}C_{U(L)}, +C_{L(U)}>) \in T_C$.
- Let $R_i = DS(s_i)$ @ P_i and let $T_i = label(R_i)$
- For each pair of consecutive transitions t_{mn} and t_{no} in the subsequence $last(DS(s_i))\ldots last(P_i)$

- If $(t_{mn}, t_{no}, <-C_{U(L)}, +C_{L(U)}>) \in T_C$ then insert the external coordination message exchange $<-C_{U(L)}, +C_{L(U)}>$ between t_{mn} and t_{no} in T_i.
- If $(t_{mn}, t_{no}, <-O_{U(L)}, +O_{L(U)}>) \in T_O$ then insert the external coordination message exchange $<-O_{U(L)}, +O_{L(U)}>$ between t_{mn} *and* t_{no} in T_i and remove $(t_{mn}, t_{no}, <-O_{U(L)}, +O_{L(U)}>)$ from T_O.

The digraph $G' = (V', E')$ is constructed to avoid choosing T-sequences which end at a state s_k where $DS(s_k)$ is very expensive, since the tail of every T-sequence must be verified by applying D. If multiple min-cost paths exist on G' then a tie-break is performed by choosing the path whose last transition causes the fewest observability and controllability problems when followed by transitions starting at $tail(R_i)$, thereby potentially reducing the number of external coordination message exchanges in the resulting checking sequence.

For $2p$-FSM $M8$, the $cost(z_{ij})$ for each dashed edge in G' is shown in Table 12, and digraph $G' = (V', E')$ is shown in Figure 24. To improve readability, only dashed edges with a cost greater than 0 are shown; there are two such edges, both of which represent $DS(s_4)$, which contains a controllability problem. As a result of Step 2, T-sequences for states s_2, s_3 and s_4 consist of only the label of $DS(s_i)$, $i = 2,3,4$. For state s_1, three min-cost paths from v_2-$t7$-v_4 to Z exist; applying either $t9$, $t10$ or $t11$ will avoid the controllability problem that is encountered if D were to be applied at s_4. The transition $t11$ is chosen as it is not the first transition in any triples in T_O or T_C, and so $T1 = label(t1\ t7\ t11)$.

Table 12. $cost(z_{ij})$ for 2p-FSM $M8$

Edge z_{ij} = $((v_i\text{-}t_{ij}\text{-}v_j), Z; <\text{-},\text{->})$	$cost(z_{ij})$
$((v_1\text{-}t1\text{-}v_3), Z; <\text{-},\text{->})$	0
$((v_1\text{-}t3\text{-}v_4), Z; <\text{-},\text{->})$	w
$((v_2\text{-}t4\text{-}v_1), Z; <\text{-},\text{->})$	0
$((v_2\text{-}t5\text{-}v_1), Z; <\text{-},\text{->})$	0
$((v_3\text{-}t6\text{-}v_1), Z; <\text{-},\text{->})$	0
$((v_3\text{-}t7\text{-}v_4), Z; <\text{-},\text{->})$	w
$((v_4\text{-}t9\text{-}v_2), Z; <\text{-},\text{->})$	0
$((v_4\text{-}t10\text{-}v_3), Z; <\text{-},\text{->})$	0
$((v_4\text{-}t11\text{-}v_2), Z; <\text{-},\text{->})$	0

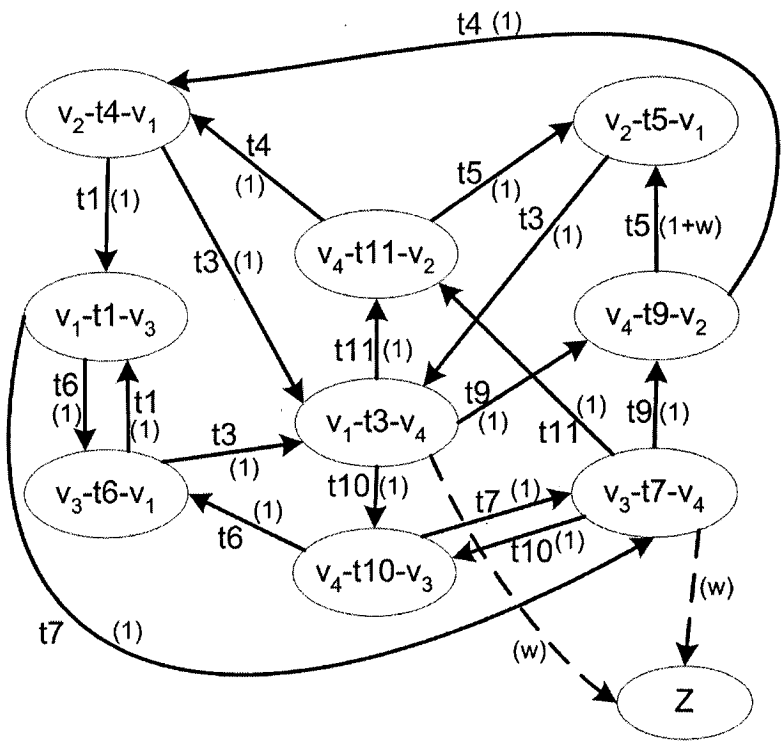


Figure 24. Digraph $G' = (V', E')$ of 2p-FSM $M8$

In Phase 4, α'-sequences are constructed from the T-sequences to verify the uniqueness of each state s_i in the implementation. Phase 4 proceeds as follows:

Let Q denote the set of states not yet recognized in some α'-sequence, and let α'_C = $label(P_C)$ be the α'-sequence currently under construction. Initially, $Q = \{s_1 \ldots s_n\}$ and $C = 1$. Perform the following steps:

Step 1. Choose some state $s_i \in Q$.

Step 2. Let $P_C = R_i$. Remove s_i from Q and let $s_j = tail(R_i)$.

Step 3. Let $old\_last = last(P_C)$.

Let $P_C = P_C@R_j$.

If $(old\_last,\ first(R_j),\ <-O_{U(L)},\ +O_{L(U)}>) \in T_O$ then insert the external coordination message exchange $<-O_{U(L)},\ +O_{L(U)}>$ between old_last and $first(R_j)$ in $label(P_C)$ and remove $(old\_last, first(R_j),\ <-O_{U(L)},\ +O_{L(U)}>)$ from T_O.

If $(old\_last, first(R_j),\ <-C_{U(L)},\ +C_{L(U)}>) \in T_C$ then insert the external coordination message exchange $<-C_{U(L)},\ +C_{L(U)}>$ between old_last and $first(R_j)$ in $label(P_C)$.

Step 4. If $s_j \in Q$ then remove s_j from Q, let $s_j = tail(R_j)$ and go to Step 3,

else (This completes P_C) let $\alpha'_C = label(P_C)$. If Q is not empty then let $C = C + 1$ and go to Step 1.

Let X_i denote the input portion of the input-output sequence T_i. The above procedure (Steps 1 through 4) ensures that every state s_i is d-recognized, and for every state s_i, $\delta(s_i, X_i)$ is d-recognized in some α'_k. Therefore $tail(R_i)$, for each R_i, is recognised in some α'_k since $tail(R_i)$ is $\delta(s_i, X_i)$. Similarly, $tail(\mathrm{P}_k)$ for each P_k is recognized since P_k ends with a subsequence X_i which is followed by the application of D in some $\mathrm{P}_{k'}$. Also, because every T-sequence and every α'-sequence begins with D, a test segment for transition t_{ij} can be formed by following $pre\_test(t_{ij})$ with either T_j, or the α'-sequence that begins at s_j (if it exists).

Applying this phase to 2p-FSM $M8$ yields three α'-sequences, α'_1 = *label*($t1$ $t7$ $t11$ $t4$ $t2$ $t1$ $t7$ $t11$), α'_2 = *label*($t6$ $t2$ $t1$ $t7$ $t11$), and α'_3 = *label*($t9$ $t5$ $t1$ $t7$ $t11$).

In Phase 5, a digraph $G'' = (V'', E'')$ is constructed in a manner similar to [HU02a]. Edges representing each *pre_test*(t_{ij}), T-sequence and α'-sequence are added to E''. A minimal symmetric augmentation $G''' = (V''', E''')$ of G[$E_\alpha \cup E_C$] is then found, and an Euler tour of G''' represents an RCPT over the subset of edges in E''' which is a synchronizable checking sequence with no potential undetectable 1-shift output faults. This phase proceeds as follows:

Step 1. Let $V'' = V \cup U'$ where $U' = \{v'_i$: for every $v_i \in V\}$

Step 2. For each edge $e_{ij} \in E$, create an edge $(v'_i, v_j$; *pre_test*(t_{ij})) in E''. Denote this set of edges E_C.

Step 3. For each T-sequence T_i = *label*(R_i), create an edge (*head*(R_i), *tail*(R_i)'; T_i) in E''. Denote this set of edges E_T.

Step 4. For each α'-sequence α'_k = *label*(P_k), create an edge (*head*(P_k), *tail*(P_k)'; α'_k) in E''. Denote this set of edges E_α.

Step 5. Add a set of edges E_S to E'' where E_S is a subset of $\{(v'_i, v'_j$; $x/y) : (v_i, v_j$; $x/y) \in E\}$ such that $G_S = (U', E_S)$ has no tour and G'' is strongly connected.

Step 6. Find the minimal symmetric augmentation $G''' = (V''', E''')$ of G[$E_\alpha \cup E_C$] by adding edges from E'' to G''.

Step 7. Find an Euler tour ET of G''', starting at v'_1, subject to the following:

Each time an edge e_k leaving a vertex v'_i is added to ET:

Let *last_trans*(e_j) denote the last transition in the transition sequence represented by e_j, where e_j is the edge immediately preceding e_k in *ET*, and let *first_trans*(e_k) denote the first transition in the transition sequence represented by e_k.

- If (*last_trans*(e_j), *first_trans*(e_k), <-$C_{U(L)}$, +$C_{L(U)}$>) $\in$ T_C then insert the external coordination message exchange <-$C_{U(L)}$, +$C_{L(U)}$> between *label*(e_j) and *label*(e_k) in *label*(*ET*).
- If (*last_trans*(e_j), *first_trans*(e_k), <-$O_{U(L)}$, +$O_{L(U)}$>) $\in$ T_O then insert the external coordination message exchange <-$O_{U(L)}$, +$O_{L(U)}$> between *label*(e_j) and *label*(e_k) in *label*(*ET*) and remove (*last_trans*(e_j), *first_trans*(e_k), <-$O_{U(L)}$, +$O_{L(U)}$>) from T_O.

The Euler tour *ET* represents an RCPT over $E_\alpha \cup E_C$ which is a checking sequence for FSM *M*.

Step 8. Eliminate any external coordination message exchanges in the resulting checking sequence that relate to potential 1-shift output faults that can be rendered detectable by some subsequence in the checking sequence, as in [LB94].

Steps 1 through 4 create the vertices V'' and add edges representing each *pre_test*(t_{ij}), T_i and α'_k. Note that each *pre_test*(t_{ij}) can only be followed by T_j or an α'_k, in any path on G''; since every T_i and α'_k begins with *D*, this forms a test segment for t_{ij}. Recall that controllability and observability problems between t_{ij} and *first*(*DS*(s_j)) are resolved as *pre_test*(t_{ij}) includes any necessary external coordination message exchange relating to controllability or observability. In Step 5, edges from *E* are added to E'' to ensure that G'' is strongly connected. Transitions which appear in the fewest triples in T_C and T_O are

chosen in this step in order to minimize potential controllability and observability problems in the resulting tour. Step 6 generates the minimal symmetric augmentation $G''' = (V''', E''')$ of $G[E_\alpha \cup E_C]$ by adding edges from E'' to G''. Step 7 then finds an Euler tour of $G''' = (V''', E''')$ which resolves potential controllability and observability problems between edges by checking for transition pairs in T_C and T_O. This check only occurs at vertices in U' as problems between t_{ij} and $first(DS(s_j))$ for each transition t_{ij} are resolved in $pre\_test(t_{ij})$ as noted above. Step 8 removes external coordination message exchanges relating to observability for any potential 1-shift output faults which are rendered detectable by some subsequence of the checking sequence.

Digraphs $G'' = (V'', E'')$ and $G''' = (V''', E''')$ for 2p-FSM $M8$ are shown in Figures 25 and 26 respectively.

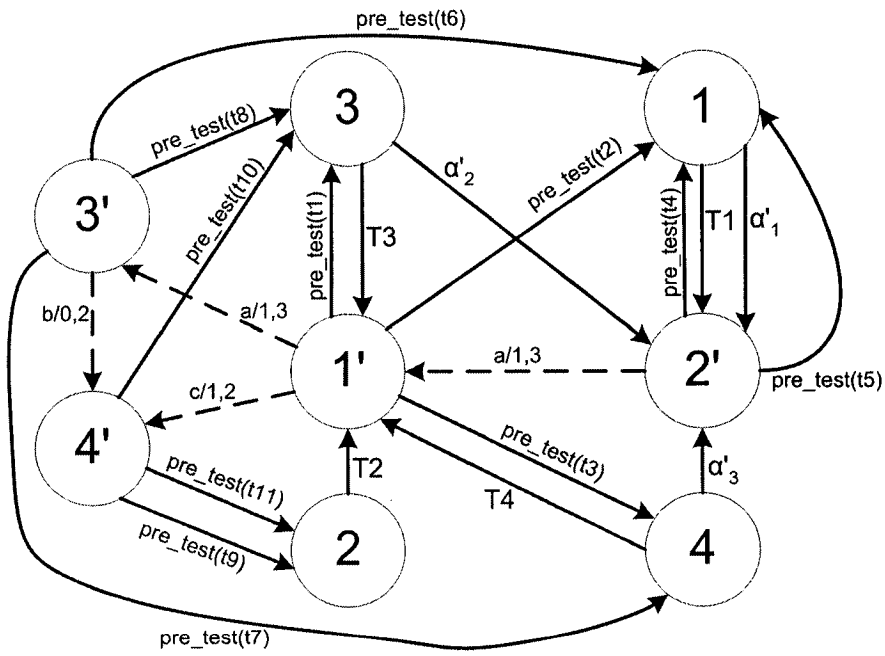


Figure 25. Digraph $G'' = (V'', E'')$ of 2p-FSM $M8$

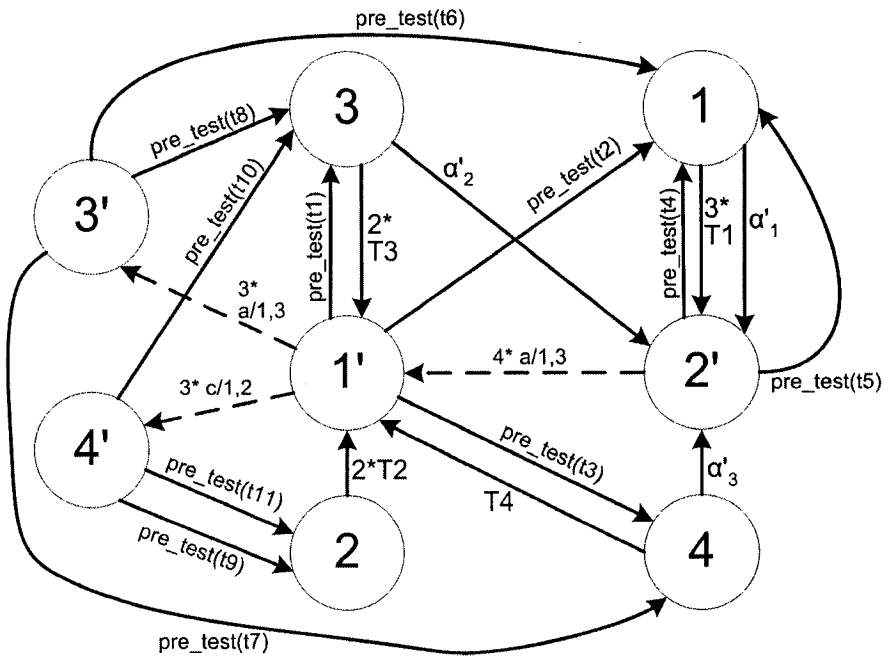


Figure 26. Digraph $G''' = (V''', E''')$ of 2p-FSM $M8$

An Euler tour of G''' starting at 1′ is: *pre_test*($t1$) α'_2 *pre_test*($t4$) α'_1 *pre_test*($t5$) $T1$ $a/1,3$ *pre_test*($t2$) $T1$ $a/1,3$ *pre_test*($t3$) α'_3 $a/1,3$ $a/1,3$ *pre_test*($t6$) $T1$ $a/1,3$ $a/1,3$ *pre_test*($t7$) $T4$ $a/1,3$ $<-O_U, +O_L>$ *pre_test*($t8$) $T3$ $c/1,2$ *pre_test*($t9$) $T2$ $c/1,2$ *pre_test*($t10$) $T3$ $c/1,2$ *pre_test*($t11$) $T2$

Note that Step 7 results in the insertion of the external coordination message exchange $<-O_U, +O_L>$ between $a/1,3$ (= $t1$) and *pre_test*($t8$) in *label*(ET). The Euler tour represents a checking sequence consisting of 58 transitions, 5 external coordination message exchanges relating to observability and 2 external coordination message exchanges relating to controllability. Step 8 results in the elimination of all 5 external coordination message exchanges relating to observability; the subsequences which render these potential 1-shift output faults detectable are shown in Table 13. The resulting checking sequence therefore requires only two external coordination message exchanges, both relating to controllability.

Table 13. Detection of potential 1-shift output faults

External Coordination Message:	Problem rendered detectable by subsequence:	Rationale
($t1$, $t8$; <-O_U, +O_L>)	$T1 = t1\ t7\ t11$	U will not receive missing '1' in faulty $t1 = a$/<-,3>
($t3$, $t9$; <-O_U, +O_L>)	*pre_test*($t7$) $T4 = t7\ t9\ t5$	U will receive extra '1' in faulty $t9 = a$/<1,2>
($t7$, $t9$; <-O_U, +O_L>)	$T1 = t1\ t7\ t11$	U will not receive missing '0' in faulty $t7 = b$/<-,2>
($t8$, $t6$; <-O_U, +O_L>)	*pre_test*($t10$) $T3 = t10\ t6\ t2$	U will not receive missing '0' in faulty $t6 = a$/<-,2>
($t9$, $t4$; <-O_U, +O_L>)	*pre_test*($t7$) $T4 = t7\ t9\ t5$	U will receive extra '1' in faulty $t9 = a$/<1,2>

A proof that checking sequences generated by this method are free of controllability problems and undetectable 1-shift output faults is given in Section 4.4.

4.4 Showing Absence of Controllability and Observability Problems

We make the claim in Sections 4.2 and 4.3 that the proposed methods yield a synchronizable checking sequence with no possibility of potential 1-shift output faults. Here we show the validity of these claims through the examination of the sequence construction methods.

Theorem 1: Given a $2p$-FSM that is minimal, deterministic and represented by a strongly connected digraph $G = (V, E)$, the method proposed in Section 4.2 constructs a checking sequence that is synchronizable and has no potential undetectable 1-shift output faults.

Proof: In the first phase of the proposed method, any two consecutive transitions on a path in $G = (V, E)$ that do not represent a pair of synchronizable transitions are added to

the set T_C. Each subsequence c_i in the checking sequence is of the form $preamble(s_i)@DS(s_i)$ (a state cover subsequence) or $preamble(s_i)@t_{ij}@DS(s_j)$ (a transition cover subsequence). Synchronization problems within each subsequence are resolved by the insertion of the corresponding external coordination message exchange relating to controllability in the following phases:

- Each $DS(s_i)$ and each $test(t_{ij}) = t_{ij}@DS(s_j)$ are made synchronizable in Phase 2, Step 1 and Step 2, respectively
- Each $preamble(s_i)$ is constructed to be synchronizable in Phase 3, Step 4
- Controllability problems between $preamble(s_i)$ and each transition t_{ij}, $head(t_{ij}) = s_i$, are resolved in Phase 4, Steps 1 and 2.

Thus every subsequence c_i in the checking sequence is synchronizable. The construction of digraph $G'' = (V'', E'')$ guarantees that any ordering of these test subsequences generated on G'' will be synchronizable. A reset input may only be sent by a tester who either sends the last input of the preceding subsequence or receives an output in the last transition of the preceding subsequence. Similarly, the first input of a subsequence can only be sent by the tester who sent the preceding reset, unless the dashed edge from 1^h to $1^{h'}$ is traversed, which results in the insertion of an external coordination message exchange relating to controllability and thus ensures the sequence is synchronizable. A checking sequence generated by the proposed method will therefore be synchronizable.

Observability problems are resolved by first constructing the set T_O of all potential 1-shift output faults, then checking whether given transition pairs appear in T_O during construction of the test subsequences and inserting external coordination message exchanges relating to observability as necessary. In a similar manner to the treatment of

controllability problems above, each subsequence $preamble(s_i)@DS(s_i)$ or $preamble(s_i)@t_{ij}@DS(s_j)$ is rendered free of undetectable 1-shift output faults.

Potential 1-shift output faults which may occur in the checking sequence between the last transition of a subsequence and the first transition of the following subsequence are now shown to be detectable. Consider two consecutive subsequences c_i and c_j in a checking sequence where the transition pair $last(c_i)$, $first(c_j)$ forms a potential 1-shift output fault. This potential 1-shift output fault is shown to be detectable as follows:

For every transition t_{ij} there exists a subsequence $preamble(s_i)@t_{ij}@DS(s_j)$. Thus there exists a subsequence $c_k = preamble(head(last(c_i)))@last(c_i)@DS(tail(last(c_i)))$ which verifies transition $last(c_i)$. If $preamble(head(last(c_i)))$ is not null then an extra or missing output in $last(c_i)$ will be detected in c_k. Similarly, if $preamble(head(first(c_j)))$ is not null then an extra or missing output in $first(c_j)$ will be detected.

We show by contradiction that an undetectable 1-shift output fault cannot occur in the checking sequence in the remaining case where both $preamble(head(last(c_i)))$ and $preamble(head(first(c_j)))$ are null:

Suppose that $preamble(head(last(c_i)))$ is null. Then $head(last(c_i)) = s_1$ since only transitions which start at the initial state have a null preamble. In order for the 1-shift output fault between $last(c_i)$ and $first(c_j)$ to remain undetected, $last(c_i)$ must always immediately precede or succeed some faulty transition t_{wx} which compensates for the extra or missing output in $last(c_i)$. However, $s_w = head(t_{wx})$ must be s_1, else the extra or missing output in t_{wx} will be detected in the subsequence $preamble(s_w)@t_{wx}@DS(s_x)$. We now have $preamble(head(last(c_i))) =$ null and $preamble(s_w) =$ null. Consider the subsequence $preamble(head(last(c_i)))@last(c_i)@DS(tail(last(c_i))$. This subsequence must

be preceded in the checking sequence by a subsequence that ends with transition t_{wx} in order for the missing or extra output in $last(c_i)$ to remain undetectable. Since every subsequence ends with the distinguishing sequence, t_{wx} must be $last(DS(s_k))$ for some state s_k. But $last(c_i)$ is also the last transition of the distinguishing sequence, and both t_{wx} and $last(c_i)$ must start at the initial state. If $head(t_{wx}) = s_1$, $head(last(c_i)) = s_1$, and the input portion of both t_{wx} and $last(c_i)$ is $last(DS)$, then $t_{wx} = last(c_i)$. Therefore the transition pair t_{wx}, $last(c_i)$ cannot form an undetectable 1-shift output fault as they are the same transition. Thus we have shown that any 1-shift output fault between the last transition of some subsequence c_i and the first transition of the following subsequence c_j will be detected by the checking sequence.

A checking sequence generated by this method would therefore be synchronizable, and would have no potential undetectable 1-shift output faults.

Q.E.D.

Theorem 2: Given a $2p$-FSM that is minimal, deterministic and represented by a strongly connected digraph $G = (V, E)$, the method proposed in Section 4.3 constructs a checking sequence that is synchronizable and has no potential undetectable 1-shift output faults.

Proof : In the first phase of the proposed method, any two consecutive transitions on a path in $G = (V, E)$ that do not represent a pair of synchronizable transitions are added to the set T_C. Similarly, any two consecutive transitions on a path in $G = (V, E)$ that form a potential 1-shift output fault are added to the set T_O. Controllability and observability problems within each $DS(s_i)$, R_i whose label is a T-sequence and P_k whose label is an α'-sequence are resolved by checking for each consecutive transition pair in T_C and T_O and

inserting the corresponding external coordination message exchanges relating to controllability and observability in the following steps of the method:

1) Problems within each $DS(s_i)$ are resolved in Phase 2, Step 1
2) Problems within each R_i are resolved in Phase 3, Step 2
3) Problems within each P_k are resolved in Phase 4, Step 3

Thus every $DS(s_i)$, R_i and P_k is free of controllability and observability problems. We now show that the resulting checking sequence is free of controllability and observability problems by examining the construction of the digraph G''' and the Euler tour.

In the digraph G''', the only edges which terminate at a node v_j (not v'_j) are those labelled $pre\_test(t_{ij})$, and the only edges starting at v_j are labelled either T_j or α'_k. Both T_j and α'_k are labels of transition sequences which start with $first(DS(s_j))$. In Phase 2, Step 2, any controllability and observability problems in the transition pair t_{ij}, $first(DS(s_j))$ are resolved by including in $pre\_test(t_{ij})$ any necessary external coordination message exchanges relating to controllability and observability. Thus any pair of consecutive edges e_m, e_n where $tail(e_m) = head(e_n) = v_j$ represent a transition sequence with no controllability or observability problems.

During the construction of the Euler tour of G''' in Phase 5, Step 7, each time a pair of consecutive edges e_p, e_q where $tail(e_p) = head(e_q) = v'_j$ is traversed, the method checks for the transition pair $last\_trans(e_p)$, $first\_trans(e_q)$ in T_C and T_O and inserts any necessary external coordination message exchanges relating to controllability and observability. Thus any pair of consecutive edges e_p, e_q where $tail(e_p) = head(e_q) = v'_j$ represent a transition sequence with no controllability or observability problems.

As a result, any transition sequence represented by a subpath crossing a node v_j or v'_j in G''' has been shown to be free of controllability and observability problems. The checking sequence represented by an Euler tour of G''' would therefore be synchronizable, and would have no potential undetectable 1-shift output faults.

Q.E.D.

4.5 Comparison with Other Methods

In [WD01] it is shown that it is not necessary to formally distinguish between forward and backward 1-shift output faults in order to properly place the external coordination message exchanges relating to observability, nor are different types of messages required for different types of output shift faults. The method proposed in this thesis for constructing the set of all potential undetectable 1-shift output faults improves upon [WD01] by considering how external coordination message exchanges relating to controllability may also resolve potential undetectable 1-shift output faults. We showed in Section 4.2 that if a transition pair t_{ij} t_{jk} forms a controllability problem then any potential 1-shift output fault involving the sender of x in t_{ij} can be resolved without the use of an external coordination message exchange relating to observability. Specifically, the sender of x in t_{ij} waits a predetermined period of time before sending the external coordination message exchange relating to controllability to the sender of input in t_{jk}. In the case of a potential backward shift fault, the sender of x in t_{ij} waits to ensure that an extra output is not received before sending the external coordination message relating to controllability. In the case of a potential forward output shift fault, the sender of x in t_{ij} waits to ensure that the expected output is received before sending the external

coordination message exchange relating to controllability. Thus our approach improves upon [WD01] by disregarding potential 1-shift output faults which are rendered detectable by this treatment of external coordination message exchanges relating to controllability.

One alternative to the method presented in Section 4.2 is to first generate a checking sequence using the *D*-method [UW97] and then identify controllability and observability problems and insert external coordination message exchanges relating to controllability and observability. In the *D*-method, the shortest transition sequence from s_1 to s_j is chosen as *preamble*(s_j). However, this may introduce controllability and/or observability problems that are avoided by our proposed method. For example, applying the *D*-method to the example FSM *M*7 from Section 4.2 yields a checking sequence composed of 10 reset inputs, 38 non-reset inputs, 1 external coordination message exchange relating to observability, and 1 external coordination message exchange relating to controllability. As a result of the controllability problem, this checking sequence requires a test architecture which supports direct communication among testers, while the checking sequence generated by our method for this same FSM does not.

Our method would tie the *D*-method in terms of the number of non-reset inputs for FSM *M*7 if triples in T_O were not removed during the construction of test segments. In that case, the preamble *t*2 *t*6 *t*9 would be chosen for s_2 instead of *t*3 *t*9; while this is a longer preamble, prefix elimination results in a checking sequence of only 38 non-reset inputs and no external coordination message exchanges relating to controllability and observability. Unfortunately, considering the effect of prefix elimination during the selection of preambles would be computationally expensive. The same remark applies to

the D-method as there may be numerous shortest paths to some states. Another computationally expensive issue stems from the possible existence of more than one distinguishing sequence for a given FSM. It is therefore recognized that our heuristic approach may not yield the shortest synchronizable checking sequence for a given FSM.

One judicious alternative to the method presented in Section 4.3 is to first generate a checking sequence using the method in [HU02a] and then identify controllability and observability problems and insert external coordination message exchanges relating to controllability and observability. It is shown in [HU02a] that the methods in [HF64, GG70, UW97] are special cases of the method in [HU02a]. Our proposed method improves upon [HU02a] in two ways. The first improvement is the selection of T-sequences. Recall that the end state of every T-sequence is verified in some α'-sequence by applying D. The method in [HU02a] does not consider controllability or observability problems that will result from a given choice of T_i. In their method, the subsequence P_i in T-sequence $T_i = label(DS(s_i)@P_i)$ is typically empty. In contrast, our method selects P_i in $T_i = label(DS(s_i)@P_i)$ as the (possibly empty) subsequence which minimizes external coordination message exchanges in the subsequence $DS(s_i)@P_i@DS(s_k)$, $s_k = tail(P_i)$. As a result, α'-sequences generated by our method will contain equal or fewer external coordination message exchanges than those selected by [HU02a].

The careful selection of T-sequences also results in possibly fewer external coordination message exchanges in the resulting checking sequence in another indirect manner. Recall that $tail(t_{ij})$ for each transition t_{ij} is verified in the checking sequence by either T_j or α'_k if such an α'-sequence exists. If $DS(s_j)$ contains controllability problems, the number of external coordination message exchanges in the resulting checking

sequence can be reduced if there is an α'-sequence α'_k which is the label of a path P_k that starts with $DS(s_j)$, as α'_k then not only verify $tail(T_j)$ but also verifies the end state of a transition t_{ij} which terminates at s_j. Our method results in T-sequences for which $cost(DS(s_j))$ is high occurring at the start of α'-sequences, thereby reducing the number of external coordination message exchanges in the checking sequence.

To demonstrate this improvement over [HU02a], consider the $2p$-FSM $M8$ from Section 4.3. T-sequences and α'-sequences chosen by [HU02a], where $T_i = label(DS(s_i))$ for all s_i, are shown in Table 14. Note that no α'-sequence begins at state s_4, the only state for which $DS(s_i)$ contains a controllability problem. This results in two occurrences of $DS(s_4)$ in the checking sequence, once in $T4$, which will be required in the checking sequence to verify transition $t7$, and once in α'_1, to verify the end state of $T1$. In contrast, our method in Section 4.3 resulted in an α'-sequence which starts at s_4, therefore requiring only one occurrence of $DS(s_4) = t9 <\text{-}C_U, +C_L> t5$ in the checking sequence.

Table 14. Sequences chosen by [HU02a] for $2p$-FSM $M8$

T- or α'-sequence	Corresponding sequence of labels
$T1$	$t1\ t7$
$T2$	$t4\ t2$
$T3$	$t6\ t2$
$T4$	$t9 <\text{-}C_U, +C_L> t5$
α'_1	$t1\ t7\ t9 <\text{-}C_U, +C_L> t5\ t1\ t7$
α'_2	$t4\ t2\ t1\ t7$
α'_3	$t6\ t2\ t1\ t7$

The second improvement to [HU02a] in our proposed method is the selection of edges in E_S. As [HU02a] does not consider controllability or observability problems, these edges are chosen arbitrarily. In contrast, our method selects transitions which do

not appear in triples in T_C or T_O whenever possible, resulting in the insertion of fewer external coordination message exchanges during the construction of the Euler tour.

For 2p-FSM $M8$, one possible set of edges selected by [HU02a] is E_S = {(v'_1, v'_3 ; a/1,3), (v'_2, v'_1 ; b/0,3), (v'_4, v'_2 ; a/-,2), (v'_4, v'_3 ; b/0,3)}. Using this set E_S and the sequences in Table 14 results in the digraph G' shown in Figure 27. The minimal symmetric augmentation G'' is shown in Figure 28. Note that including the edge (v'_4, v'_2 ; a/-,2) in E_S will result in the insertion of an external coordination message exchange relating to controllability, whereas using (v'_4, v'_2 ; c/1,3) would not.

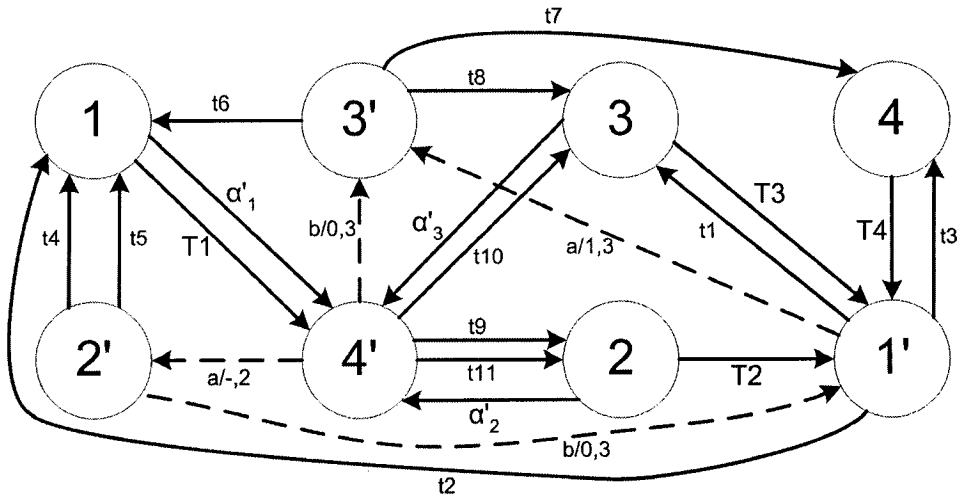


Figure 27. Digraph $G' = (V', E')$ of 2p-FSM $M8$ by [HU02a]

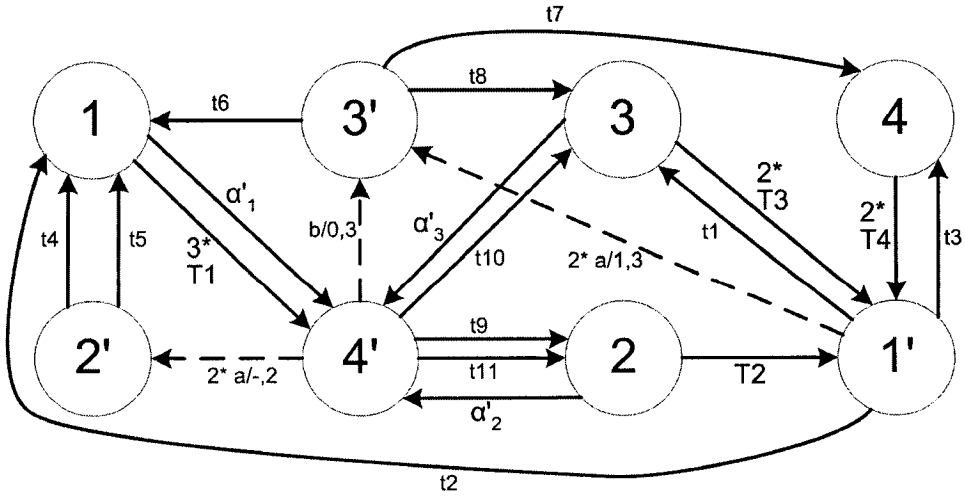


Figure 28. Digraph $G'' = (V'', E'')$ of 2p-FSM $M8$ by [HU02a]

An Euler tour of the digraph G'' in Figure 28 is:

$t2\ \alpha'_1\ t10\ T3\ t3\ T4\ t1\ \alpha'_3\ a/\text{-},2\ t5\ T1\ a/\text{-},2\ t4\ T1\ t9\ \alpha'_2\ t11\ T2\ a/1,3\ t6\ T1\ b/0,3\ t8\ T3\ t1\ T3\ a/1,3\ t7,\ T4$

Given the sets T_C and T_O from Section 4.3, external coordination message exchanges relating to controllability and observability are inserted into the corresponding checking sequence, as in [CR99]. This yields a checking sequence consisting of 47 transitions, 4 external coordination message exchanges relating to controllability, and 5 external coordination message exchanges relating to observability, ie. the input portion of the sequence:

$t2\ t1\ t7 <\text{-}O_U, +O_L> t9 <\text{-}C_U, +C_L> t5\ t10\ t6\ t2\ t3 <\text{-}O_U, +O_L> t9 <\text{-}C_U, +C_L> t5\ t1\ t6\ t2\ t1\ t7\ t9 <\text{-}C_U, +C_L> t5\ t1\ t7\ t9 <\text{-}O_U, +O_L> t4\ t1\ t7\ t9\ t4\ t2\ t1\ t7\ t11\ t4\ t2\ t1\ t6\ t1\ t7\ t10 <\text{-}O_U, +O_L> t8 <\text{-}O_U, +O_L> t6\ t2\ t1\ t6\ t2\ t1\ t7\ t9 <\text{-}C_U, +C_L> t5$

As in the proposed method in Section 4.3, closer inspection of the checking sequence reveals that all external coordination message exchanges relating to observability can be removed as these potential 1-shift output faults are rendered detectable by other subsequences in the checking sequence. However, the checking sequence obtained by [HU02a] contains 4 external coordination message exchanges relating to controllability, compared to only 2 in the checking sequence generated by our method.

4.6 Extending the Proposed Methods for *np*-FSMs

The proposed methods are generalized to accommodate *np*-FSMs, where $n > 2$. Recall the definition of *np*-FSM given in Section 2.1 and consider the digraph $G = (V, E)$ of an *np*-FSM M, where n testers interacting with the IUT are labelled 1, 2, 3, ... , n. We

consider first the construction of sets T_C and T_O in the case of n testers, followed by the extension of the two proposed methods such that each can be applied to yield a checking sequence for an *np*-FSM.

4.6.1 Controllability and Observability Problems in *np*-FSMs

In both methods proposed in Section 4.2 and 4.3, the first steps are to find the set of all controllability problems T_C and the set of all observability problems T_O. These steps are adapted to the general case as follows:

Step 1. The set T_C is constructed from $G = (V, E)$ as follows:

for each vertex $v_j \in V$ *do*

for each edge e_{ij} *(say* t_{ij}*)* $= (v_i, v_j; x_j/y_j)$ *entering vertex* v_j *do*

for each edge e_{jk} *(say* t_{jk}*)*$= (v_j, v_k; x_{j+1}/y_{j+1})$ *leaving vertex* v_j *do*

if $x_j \in \Sigma_q$ *AND* $x_{j+1} \in \Sigma_r$ *AND* $a_r =$ *- in* y_j

then add $(t_{ij}, t_{jk}; <-C_r, +C_q>)$ *to* T_C

where $1 \leq r, q \leq n$

Step 2. The set T_O is constructed from $G = (V, E)$ as follows:

for each vertex $v_j \in V$ *do*

for each edge e_{ij} *(say* t_{ij}*)* $= (v_i, v_j; x_j/y_j)$ *entering vertex* v_j *do*

for each edge e_{jk} *(say* t_{jk}*)*$= (v_j, v_k; x_{j+1}/y_{j+1})$ *leaving vertex* v_j *do*

if for some output $a_q \in \Gamma_q$, a_q *is in* y_j *XOR* a_q *is in* y_{j+1} *AND* $x_{j+1} \notin \Sigma_q$

AND $(t_{ij}, t_{jk}; <-C_r, +C_q>) \notin T_C$

then add $(t_{ij}, t_{jk}, <-O_q, +O_r>)$ *to* T_O,

where r is the tester sending the input x_{j+1} in t_{jk} and q is the tester involved in the shift.

Consider the digraph $G = (V, E)$ of the $3p$-FSM $M9$ shown in Figure 27, where the three testers interacting with the IUT are labeled 1, 2, and 3. The sets T_C and T_O are found to be:

$T_C = \{(t7, t3, <-C_3, +C_1>)\}$

$T_O = \{(t2, t7, <-O_3, +O_1>)\ (t4, t7, <-O_3, +O_1>), (t5, t7, <-O_3, +O_1>), (t7, t1, <-O_3, +O_1>),$

$(t7, t2, <-O_3, +O_2>)\}$

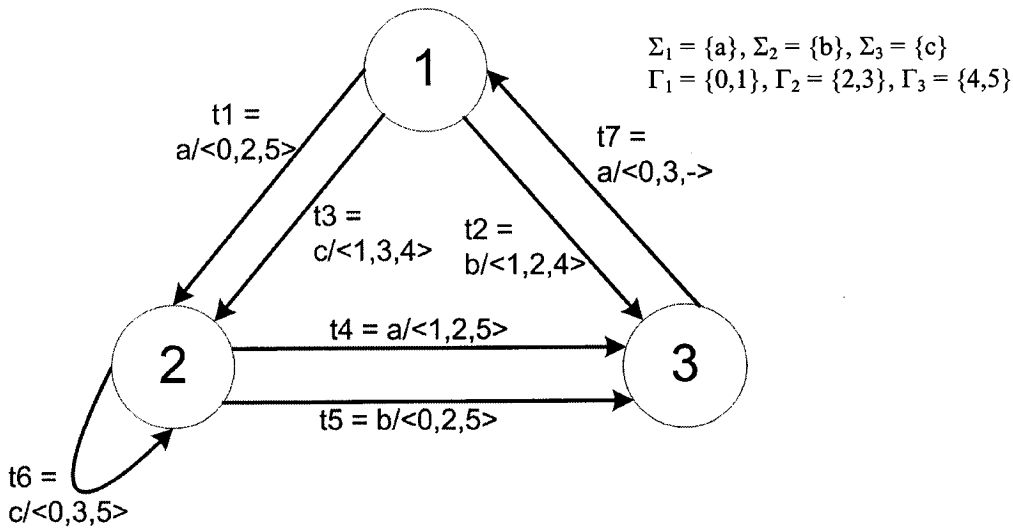


Figure 29. Digraph $G = (V, E)$ of $3p$-FSM $M9$

A given transition pair $t_{ij}\ t_{jk}$ may appear in triples in both T_C and T_O. For example, given the input and output alphabets in Figure 27, consider the consecutive transition pair $t_{ij} = (s_i, s_j; a/<1,-,->)$ and $t_{jk} = (s_j, s_k; b/<0,2,4>)$. This pair requires both an external coordination message $<-C_2, +C_1>$ relating to controllability and $<-O_3, +O_2>$ relating to observability. In this case, the external coordination message exchange relating to controllability must precede any external coordination message exchanges relating to observability in the checking sequence. For the pair $t_{ij}\ t_{jk}$, Tester 2 would therefore

receive an external coordination message exchange relating to controllability from Tester 1 and then send an external coordination message exchange relating to observability to Tester 3. Note also that a given transition pair may cause more than one potential 1-shift output fault, which results in the insertion of more than one external coordination message exchange relating to observability.

4.6.2 Reliable Reset Method for *np*-FSMs

Given the sets T_C and T_O, the method proposed for $2p$-FSMs in Section 4.2 forms a checking sequence based on the use of a reliable reset. This method is adapted for *np*-FSMs as follows.

First, problems relating to controllability and observability within the transition sequence $DS(s_i)$ induced by D at state s_i are resolved by inserting corresponding external coordination message exchanges from T_C and T_O. This step proceeds similar to the case of $n = 2$ testers, given in Section 4.2. Formally, this phase consists of two steps:

Step 1. For each state s_i, find the transition sequence $DS(s_i)$ induced by D on G at v_i.

For each pair of consecutive transitions $t_{mn}\ t_{no}$ in $DS(s_i)$:

- If $(t_{mn}, t_{no}, <-O_q, +O_r>) \in T_O$ then insert the external coordination message exchange $<-O_q, +O_r>$ between t_{mn} and t_{no} in $label(DS(s_i))$ and remove $(t_{mn}, t_{no}, <-O_q, +O_r>)$ from T_O.
- If $(t_{mn}, t_{no}, <-C_r, +C_p>) \in T_C$ then insert the external coordination message exchange $<-C_r, +C_p>$ between t_{mn} and t_{no} in $label(DS(s_i))$.

Step 2. For each transition $t_{ij} = (v_i, v_j; x_j/y_j)$:

- Construct $test(t_{ij}) = t_{ij}@DS(s_j)$.

- If there is a triple (t_{mn}, t_{no}, <-O_q, +O_r>) $\in$ T_O where t_{mn} = t_{ij} and t_{no} = *first*($DS(s_j)$) then insert the external coordination message exchange(s) <-O_q, +O_r> between t_{ij} and $DS(s_j)$ in *label*(*test*(t_{ij})) and remove (t_{mn}, t_{no}, <-O_q, +O_r>) from T_O.
- If there is a triple (t_{mn}, t_{no}, <-C_r, +C_p>) $\in$ T_C where t_{mn} = t_{ij} and t_{no} = *first*($DS(s_j)$) then insert the external coordination message exchange <-C_r, +C_p> between t_{ij} and $DS(s_j)$ in *label*(*test*(t_{ij})).

Note that in both steps 1 and 2, a given transition pair t_{mn} t_{no} may appear in more than one triple in T_O. In this case, all corresponding external coordination message exchanges relating to observability must be inserted between t_{mn} and t_{no}, and all elements with the pair t_{mn} t_{no} are removed from T_O.

For the 3*p*-FSM *M*9, the distinguishing sequence is $D = a$. As the distinguishing sequence is a single transition, Step 1 is not used. The test segments constructed in Step 2 are shown in Table 15. Note that this step removes 4 triples from T_O, resulting in T_O = {(*t*7, *t*2, <-O_3, +O_2>)}.

Table 15. Test segments for 3*p*-FSM *M*9

Transition t_{ij}	*label*(*test*(t_{ij}))
*t*1	*t*1 *t*4
*t*2	*t*2 <-O_3, +O_1> *t*7
*t*3	*t*3 *t*4
*t*4	*t*4 <-O_3, +O_1> *t*7
*t*5	*t*5 <-O_3, +O_1> *t*7
*t*6	*t*6 *t*4
*t*7	*t*7 <-O_3, +O_1> *t*1

The *sum_cost* computation at Step 1 of Phase 3 of Section 4.2 is applied in the case of *n* testers, with the only change being the replacement of the subscripts *U*(*L*) and *L*(*U*) with the general subscripts *q* and *r*. The *sum_cost* values for 3*p*-FSM *M*9 are shown in

Table 16. Phase 3 of the method presented in Section 4.2 proceeds with the construction of graph $G' = (V', E')$, and a graph $G_j = (V_j, E_j)$ for each state $s_j \neq s_1$. Subscripts $U(L)$ and $L(U)$ are again replaced with the general subscripts q and r. The preamble for state s_i is then selected as the label of the min-cost path found on graph G_i.

Table 16. *sum_cost*(t_{ij}) for transition t_{ij}

Transition t_{ij}	*sum_cost*(t_{ij})
$t1$	0
$t2$	0
$t3$	0
$t4$	0
$t5$	0

Note that the only potential controllability problem for 3p-FSM $M9$ is ($t7$, $t3$, <-C_3, +C_1>) and the only remaining potential 1-shift output fault is ($t7$, $t2$, <-O_3, +O_2>). Since the first transition in both triples is $t7$, and *tail*($t7$) = s_1, preambles for s_2 and s_3 can be chosen as the transition sequence represented by the shortest path on $G = (V, E)$ without constructing $G' = (V', E')$. Thus we select *preamble*(s_2) = $t1$ and *preamble*(s_3) = $t2$.

State and transition covers are then formed and prefix elimination is applied according to Steps 1 through 4 of the final phase of the method presented in Section 4.2. Resulting subsequences are given in Table 17. To generate a synchronizable ordering of the remaining subsequences, the graph $G'' = (V'', E'')$ is constructed similar to Steps 5 through 10 of the final phase of the method given in Section 4.2 as follows:

In the following steps, h is the tester sending the *last* input of D.

Step 1. For each tester q, $1 \leq q \leq n$, create a vertex in V'' labelled 1^q.

Step 2. Create a vertex T^h in V''.

Step 3. For each sequence $c_i \in C$

- Let $\{m, n, \ldots, q\}$ be the set of all testers (other than h) who receive output in the transition $last(c_i)$.
- Create a vertex in V'' labelled $T^{h, m, n, \ldots, q}$ if one does not exist already.
- Add a solid edge $(1^k, T^{h, m, n, \ldots, q}; l_i)$ to E'', where tester k is the sender of x in $first(c_i)$ and $l_i = label(c_i)$.

Step 4. Add the following dashed edges to E'' representing reset transitions:

For each vertex $T^{h, m, n, \ldots, q}$ in V''

For each $k \in \{h, m, n, \ldots, q\}$

create a dashed edge ($T^{h, m, n, \ldots, q}$, 1^k; "Rf*k*/- ")

Step 5. Create a vertex in V'' labelled "*Start*". For each vertex 1^k, $k \neq h$, add a dashed edge (1^h, 1^k; $<-C_k, +C_h>$) and a dashed edge (*Start*, 1^k ; -) to E''.

After this step is complete, the resulting digraph will be known as $G'' = (V'', E'')$.

Step 6. Beginning at *Start*, find a rural Chinese postman path (RCPP) over the solid edges in E''. The input portion of the label of this path represents a checking sequence for FSM M.

Step 7. Eliminate any external coordination message exchanges in the resulting checking sequence that relate to potential 1-shift output faults that can be rendered detectable by some subsequence in the checking sequence, as in [LB94].

For 3p-FSM $M9$, the 5 sequences remaining after prefix elimination are shown in Table 17.

Table 17. Checking sequence subsequences for 3*p*-FSM *M*9

Label	Transition Sequence	Verifies
L_1	$t3\ t4$	$t3$
L_2	$t1\ t4 <\text{-}O_3, +O_1> t7$	$s_2, t1, t4$
L_3	$t1\ t5 <\text{-}O_3, +O_1> t7$	$t5$
L_4	$t1\ t6\ t4$	$t6$
l_5	$t2 <\text{-}O_3, +O_1> t7 <\text{-}O_3, +O_1> t1$	$s_3, t2, t7$

The graph $G'' = (V'', E'')$ for $M9$ is shown in Figure 30. The vertex *Start* is added to V'' in the case of an *np*-FSM as the checking sequence may start with an input from any tester other than the sender of the last input of D. Dashed edges representing external coordination message exchanges relating to controllability are added to ensure that a rural Chinese postman path over the solid edges exists. For FSM $M9$, one such path is: l_1 rf2/- l_5 rf1/- l_4 rf1/- l_2 rf1/- l_3,

which corresponds to the sequence

$t3\ \ t4$ rf2/- $t2 <\text{-}O_3, +O_1> t7 <\text{-}O_3, +O_1> t1$ rf1/- $t1\ t6\ t4$ rf1/- $t1\ t4 <\text{-}O_3, +O_1> t7$ rf1/- $t1$ $t5 <\text{-}O_3, +O_1> t7$

The input portion of this sequence is a checking sequence with no undetectable 1-shift output faults.

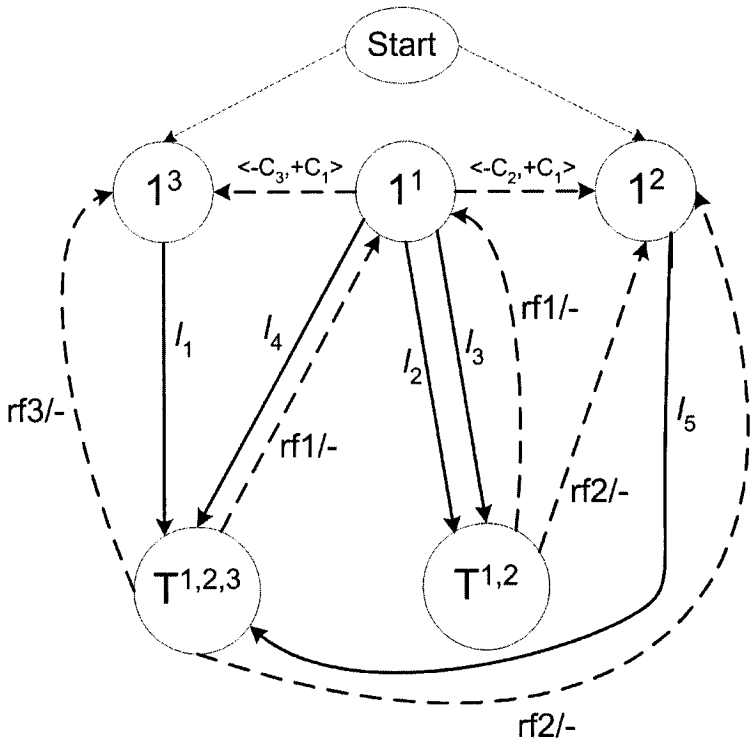


Figure 30. Digraph $G'' = (V'', E'')$ of 3*p*-FSM $M9$

Closer inspection as in [LB94] reveals that all 4 external coordination message exchanges relating to observability can be eliminated, resulting in a synchronizable checking sequence which requires no external coordination message exchanges relating to controllability or observability. The subsequences which render these potential 1-shift output faults detectable are shown in Table 18.

Table 18. Detection of potential 1-shift output faults

External Coordination Message:	Problem rendered detectable by subsequence:	Rationale
($t2$, $t7$, <-O_3, +O_1>)	$l_3 = t1\ t5\ t7$	Tester 3 receives '4' as the final output in the CS from faulty $t7 = a$/<0,3,4>
($t4$, $t7$, <-O_3, +O_1>)	l_1 rf2/- $l_5 = t3\ t4$ rf2/- $t2$	Tester 3 does not receive missing '5' in faulty $t4 = a$/<1,2,-> before it receives output '4' in $t2$

($t5$, $t7$, <-O_3, +O_1>)	l_5 rfl/- l_4 = $t2$ $t7$ $t1$ rfl/- $t1$ $t6$ $t4$	Tester 3 receives extra '5' in faulty $t7$ = a/<0,3,5>
($t7$, $t1$, <-O_3, +O_1>)	l_5 rfl/- l_4 = $t2$ $t7$ $t1$ rfl/- $t1$ $t6$ $t4$	Tester 3 does not receive missing '5' in second faulty $t1$ = a/<0,2,->

4.6.3 α'-sequences for *np*-FSMs

Given the sets T_C and T_O, the method presented in Section 4.3 constructs a checking sequence for a $2p$-FSM without requiring the use of a reliable reset. This method is extended for *np*-FSMs as follows.

First, problems relating to controllability and observability within the transition sequence $DS(s_i)$ induced by D at state s_i are resolved by inserting corresponding external coordination message exchanges from T_C and T_O. Problems between each transition t_{ij} and $DS(s_j)$ are similarly resolved. A cost is associated with each $DS(s_i)$ based on the number of controllability problems it contains. Formally, this phase consists of two steps:

Step 1. For each state s_i, find the transition sequence $DS(s_i)$ induced by D on G at v_i.

Initially, $cost(DS(s_i)) = 0$.

For each pair of consecutive transitions t_{mn} t_{no} in $DS(s_i)$:

- If (t_{mn}, t_{no}, <-O_q, +O_r>) $\in$ T_O then insert the external coordination message exchange <-O_q, +O_r> between t_{mn} and t_{no} in $label(DS(s_i))$ and remove (t_{mn}, t_{no}, <-O_q, +O_r>) from T_O.

- If $(t_{mn}, t_{no}, <-C_r, +C_p>) \in T_C$ then insert the external coordination message exchange $<-C_r, +C_p>$ between t_{mn} and t_{no} in $label(DS(s_i))$ and let $cost(DS(s_i)) = cost(DS(s_i)) + w$.

Step 2. For each edge e_{ij} (*say* t_{ij}) = $(v_i, v_j; x_j/y_j)$ in E

Let $pre\_test(t_{ij}) = label(t_{ij})$

If $(t_{ij}, first(DS(s_j)), <-C_r, +C_p>) \in T_C$

then $pre\_test(t_{ij}) = pre\_test(t_{ij}) <-C_r, +C_p>$

If $(t_{ij}, first(DS(s_j)), <-O_q, +O_r>) \in T_O$

then $pre\_test(t_{ij}) = pre\_test(t_{ij}) <-O_q, +O_r>$

remove $(t_{ij}, first(DS(s_j)), <-O_q, +O_r>)$ from T_O

Note that in the case of n testers, $n > 2$, there may be more than one external coordination message exchange inserted between a given transition pair in both Steps 1 and 2. For the 3p-FSM $M9$ shown in Figure 29, the distinguishing sequence is $D = a$. As the distinguishing sequence is a single transition, $cost(DS(s_i)) = 0$ for every state s_i. The $pre\_test(t_{ij})$ for each transition t_{ij} is shown in Table 19. Note that this step removes 4 triples from T_O, resulting in $T_O = \{(t7, t2, <-O_3, +O_2>)\}$.

Table 19. *pre_test*(t_{ij}) for 3p-FSM $M9$

Transition t_{ij}	$pre\_test(t_{ij})$
$t1$	$t1$
$t2$	$t2 <-O_3, +O_1>$
$t3$	$t3$
$t4$	$t4 <-O_3, +O_1>$
$t5$	$t5 <-O_3, +O_1>$
$t6$	$t6$
$t7$	$t7 <-O_3, +O_1>$

In the next phase, digraph G' is constructed to aid in generating T-sequences and α'-sequences, similar to the method in Section 4.3. This phase consists of 2 steps:

Step 1. Construct the digraph $G' = (V', E')$ from $G = (V, E)$ as follows:

for each vertex $v_j \in V$

for each edge e_{ij} *(say* t_{ij}*)* $= (v_i, v_j; x_j/y_j)$ *entering vertex* v_j *in* V, $head(e_{ij}) \neq v_j$:

for each edge e_{jk} *(say* t_{jk}*)* $= (v_j, v_k; x_k/y_k)$ *leaving vertex* $v_j \in V$, $tail(e_{jk}) \neq v_j$:

create a vertex labelled "v_i-t_{ij}-v_j*" in* V' *if one does not exist already*

create a vertex labelled "v_j-t_{jk}-v_k*" in* V' *if one does not exist already*

add an edge $e'_{jk} = (v_i\text{-}t_{ij}\text{-}v_j,\ v_j\text{-}t_{jk}\text{-}v_k;\ t_{jk})$ *in* E'

$cost(e'_{jk}) = 1$

for each $(t_{ij}, t_{jk}, <\text{-}O_q, +O_r>) \in T_O$ *AND* $(t_{ij}, t_{jk}, <\text{-}C_r, +C_p>) \in T_C$

$cost(e'_{jk}) = cost(e'_{jk}) + w$

create a vertex labeled Z *in* V'

for each vertex $(v_i\text{-}t_{ij}\text{-}v_j)$ *in* V'

create a dashed edge $z_{ij} = ((v_i\text{-}t_{ij}\text{-}v_j), Z; <\text{-},\ldots,\text{->})$

if $(t_{ij}, first(DS(s_j)), <\text{-}C_r, +C_p>) \in T_C$ *then* $cost(z_{ij}) = cost(DS(s_j)) + w$

else $cost(z_{ij}) = cost(DS(s_j))$

Step 2. For each state s_i, the T-sequence T_i is selected as follows:

- On G', find the min-cost path P_i from $(head(last(DS(s_i)))\text{-}last(DS(s_i))\text{-}tail(last(DS(s_i))))$ to Z.
- If there is more than one such min-cost path, P_i is chosen as the path for which the last transition in the path occurs least often as transition t_{jk} in elements $(t_{jk}, t_{kl}, <\text{-}O_q, +O_r>) \in T_O$ and $(t_{jk}, t_{kl}, <\text{-}C_r, +C_p>) \in T_C$.
- Let $R_i = DS(s_i)$ @ P_i and let $T_i = label(R_i)$

- For each consecutive transition pair t_{mn} t_{no} in the subsequence $last(DS(s_i))\ldots last(P_i)$
  - If $(t_{mn}, t_{no}, <-C_r, +C_p>) \in T_C$ then insert the external coordination message exchange $<-C_r, +C_p>$ between t_{mn} and t_{no} in T_i.
  - If $(t_{mn}, t_{no}, <-O_q, +O_r>) \in T_O$ then insert the external coordination message exchange $<-O_q, +O_r>$ between t_{mn} and t_{no} in T_i and remove $(t_{mn}, t_{no}, <-O_q, +O_r>)$ from T_O.

Digraph $G' = (V', E')$ for FSM $M9$ is shown in Figure 31. As a result of Step 2, the T-sequence chosen is $label(DS(s_i))$ for every state s_i.

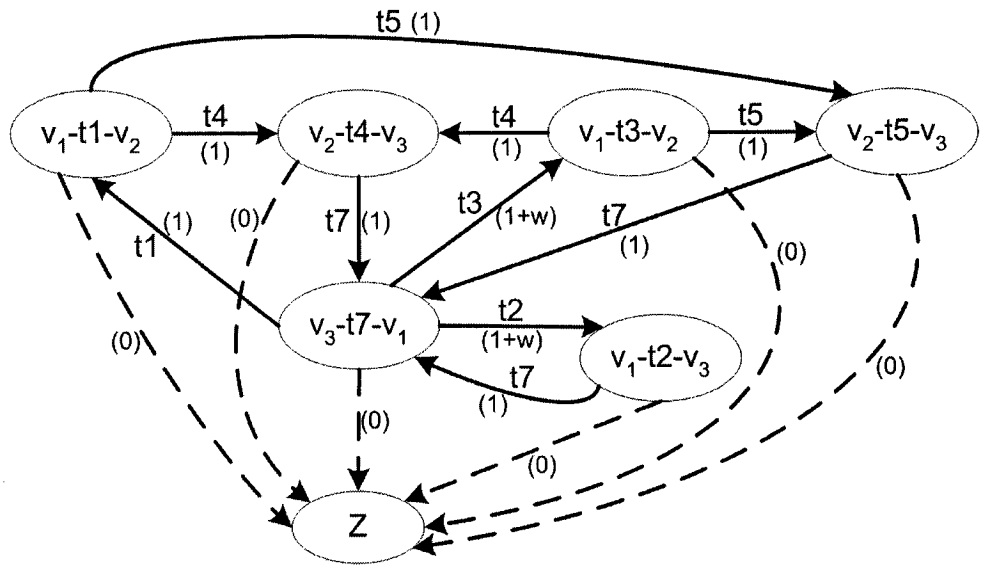


Figure 31. Digraph $G' = (V', E')$ of 3p-FSM $M9$

In the case of n testers, α'-sequences are constructed from the T-sequences similar to the method given in Phase 4 in Section 4.3. The only change required in this step is again the generalization of subscripts and the possible insertion of more than one external coordination message exchange between T-sequences during the formation of the α'-sequences. For 3p-FSM $M9$, this step results in one α'-sequence, $\alpha'_1 = label(t1\ t4\ t7\ t1)$.

Phase 5 proceeds similarly to the method for $2p$-FSMs in Section 4.3; digraph G'' and the minimal symmetric augmentation $G''' = (V''', E''')$ of $G[E_\alpha \cup E_C]$ are generated, with an Euler tour of G''' representing a RCPT over $E_\alpha \cup E_C$ in E'' which is a checking sequence for M. For an np-FSM, $n > 2$, the check which occurs each time an edge e_k leaving a vertex v'_i is added to ET in Phase 5, Step 7 in Section 4.3 may result in the insertion of more than one external coordination message exchange.

For $3p$-FSM $M9$, edges in E_S are selected as $E_S = \{(v'_1, v'_2 ; a/<0,2,5>), (v'_3, v'_1 ; a/<0,3,->)\}$. Digraphs G'' and G''' are shown in Figure 32 and Figure 33, respectively.

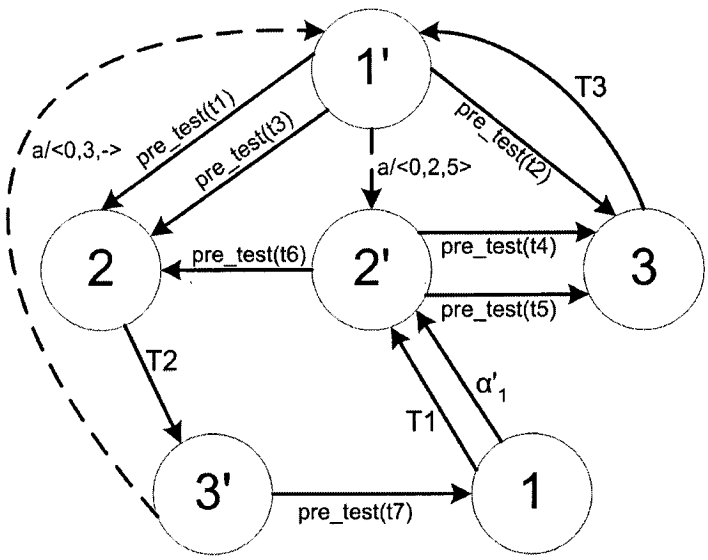


Figure 32. Digraph $G'' = (V'', E'')$of $3p$-FSM $M9$

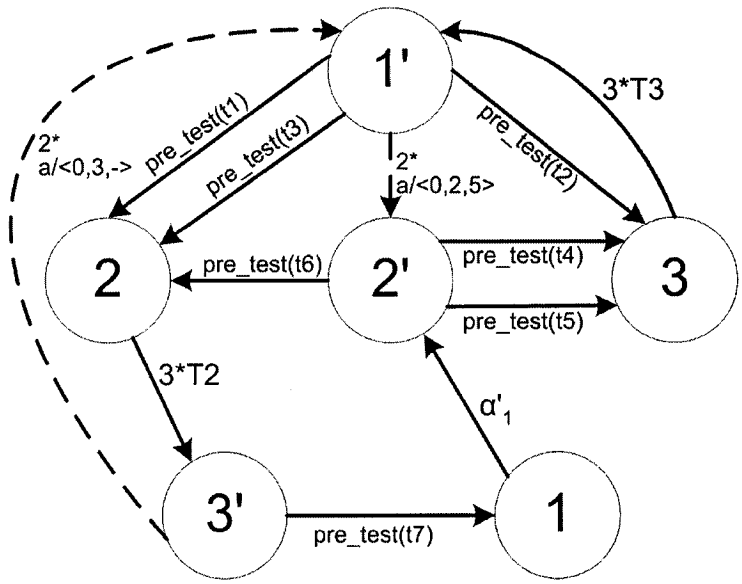


Figure 33. Digraph $G''' = (V''', E''')$ of 3p-FSM $M9$

An Euler tour of G''' starting at v'_1 is: *pre_test*($t3$) $T2$ *pre_test*($t7$) α'_1 *pre_test*($t4$) $T3$ *pre_test*($t2$) $T3$ *pre_test*($t1$) $T2$ $a/<0,3,->$ $a/<0,2,5>$ *pre_test*($t5$) $T3$ $a/<0,2,5>$ *pre_test*($t6$) $T2$ $a/<0,3,->$

This corresponds to a checking sequence of 21 transitions, 0 external coordination message exchanges relating to controllability, and 5 external coordination message exchanges relating to observability, ie. the input portion of the transition sequence:

$t3$ $t4$ $t7$ $<-O_3, +O_1>$ $t1$ $t4$ $t7$ $t1$ $t4$ $<-O_3, +O_1>$ $t7$ $<-O_3, +O_2>$ $t2$ $<-O_3, +O_1>$ $t7$ $t1$ $t4$ $t7$ $t1$ $t5$ $<-O_3, +O_1>$ $t7$ $t1$ $t6$ $t4$ $t7$

Note that Step 7 results in the insertion of the external coordination message exchange $<-O_3, +O_2>$ between $t7$ and $t2$. Step 8 eliminates all 5 external coordination message exchanges relating to observability as shown in Table 20. Thus the resulting checking sequence requires no external coordination message exchanges among the three testers.

Table 20. Detection of potential 1-shift output faults

External Coordination Message:	Problem rendered detectable by subsequence:	Rationale
($t2$, $t7$, <-O_3, +O_1>)	$\alpha'_1 = t1\ t4\ t7\ t1$	Tester 3 receives '4' in faulty $t7$ = a/<0,3,4> before receiving '5' in $t1$
($t4$, $t7$, <-O_3, +O_1>)	*pre_test*($t2$) $T3$ *pre_test*($t1$) $T2$ a/<0,3,-> a/<0,2,5> *pre_test*($t5$) $T3$ a/<0,2,5> *pre_test*($t6$) = $t2\ t7\ t1\ t4\ t7\ t1\ t5\ t7\ t1\ t6$	Possible faulty $t7$ = a/<0,3,5> is already detected (see rationale for ($t5$, $t7$, <-O_3, +O_1>) below). Therefore if faulty $t4$ = a/<1,2,-> then Tester 3 will receive only four '5's after receiving '4' in $t2$
($t5$, $t7$, <-O_3, +O_1>)	*pre_test*($t3$) $T2$ *pre_test*($t7$) α'_1 *pre_test*($t4$) $T3$ *pre_test*($t2$) = $t3\ t4\ t7\ t1\ t4\ t7\ t1\ t4\ t7\ t2$	Tester 3 receives three too many '5's before receiving '4' in $t2$, due to faulty $t7$ = a/<0,3,5>
($t7$, $t1$, <-O_3, +O_1>)	*pre_test*($t3$) $T2$ *pre_test*($t7$) α'_1 *pre_test*($t4$) $T3$ *pre_test*($t2$) = $t3\ t4\ t7\ t1\ t4\ t7\ t1\ t4\ t7\ t2$	If faulty $t1$ = a/<0,2,-> then Tester 3 will receive one too many '5's before receiving '4' in $t2$; If both faulty $t1$ = a/<0,2,-> and $t4$ = a/<1,2,-> then Tester 3 will receive only three '5's before receiving '4' in $t2$
($t7$, $t2$, <-O_3, +O_2>)	$\alpha'_1 = t1\ t4\ t7\ t1$	Tester 3 receives '4' in faulty $t7$ = a/<0,3,4> before receiving '5' in $t1$

Chapter Five

Conclusions

5.1 Final Remarks

Two general approaches can be considered in constructing test sequences and checking sequences for a distributed test architecture. In the first general approach, first a known sequence generation method is applied to the given FSM, and then the resulting sequence is examined for controllability and observability problems and corrective actions are taken. These corrective actions depend on whether the distributed test architecture supports direct communication between testers through the use of external coordination message exchanges, or only indirect communication between testers via their interactions with the IUT. In the second general approach, controllability and observability problems are considered during the construction of the sequence, thereby minimizing the number of corrective actions required to eliminate controllability and observability problems. Proposed methods in the existing literature that use this approach are currently limited to only test sequences, with the exception of [HU02b], which makes strong assumptions regarding the existence of synchronizable UIO sequences as testers are assumed to communicate only via their interactions with the IUT.

We have studied the problem of controllability and observability in distributed testing, proposing two heuristic methods for generating synchronizable checking sequences with no potential undetectable 1-shift output faults. The proposed methods attempt to minimize the use of external coordination message exchanges by considering potential problems relating to controllability and observability during the construction of the checking sequence.

5.2 Summary of Contributions

Below we list the major contributions of the thesis:

We have proposed two methods for generating a synchronizable checking sequence with no possibility of potential undetectable 1-shift output faults when applied to a $2p$-FSM in a distributed test architecture.

We have proven that, given a $2p$-FSM that is minimal, deterministic and whose underlying graph is strongly connected, the proposed methods construct a synchronizable checking sequence with no potential undetectable 1-shift output faults.

We have shown the proposed methods to perform at least as well as other methods proposed in the literature, and we have shown that they perform better than each one under certain conditions.

We have extended the proposed methods so that they can be applied to an np-FSM in a distributed test architecture, $n > 2$.

5.3 Directions for Future Research

It would be interesting to see this work improved and/or extended in the following directions:

The methods proposed in this thesis require the existence of a distinguishing sequence for the given FSM. It would be interesting to see the proposed methods adapted to use UIOs or characterizing sets for state identification, resulting in applicability to FSMs which do not have a distinguishing sequence. Also, only the prefix of a distinguishing sequence might be used in identifying some states, potentially resulting in a shorter

checking sequence and/or fewer external coordination message exchanges relating to controllability and observability.

In both proposed methods, the last step checks for external coordination message exchanges which are redundant as the potential 1-shift output fault is rendered detectable by some other subsequence, as discussed in [LB94]. It may be advantageous to include this type of check at an earlier stage in the proposed methods so the methods will not unnecessarily avoid transition pairs that form an observability problem that will be detected by an existing subsequence.

In our proposed solution based on α'-sequences, it may be advantageous to also consider the impact on the connectivity of the resulting graph G'' during the selection of T-sequences, thus possibly reducing the length of the resulting checking sequence.

The proposed solutions consider only potential undetectable 1-shift faults. A potential undetectable k-shift output fault occurs when in any two transitions labeled x_j/y_j and x_{j+k}/y_{j+k}, $k \geq 2$, these two transitions satisfy the conditions for an undetectable 1-shift output fault and the k-shift output fault is not detected by the testers involved in the intermediate transitions between x_j/y_j and x_{j+k}/y_{j+k}. It would be interesting to find sufficient constraints or conditions to guarantee that any potential undetectable k-shift output fault is rendered detectable by the proposed methods.

REFERENCES

[BU91] S. Boyd and H. Ural, "The synchronization problem in protocol testing and its complexity," *Information Processing Letters*, vol. 40, pp. 131-136, 1991.

[CR99] L. Cacciari and O. Rafiq, "Controllability and Observability in Distributed Testing," *Information and Software Technology*, vol. 41, pp. 767-780, 1999.

[CT78] T.S. Chow, "Test software design modelled by finite state machines," *IEEE Transactions on Software Engineering*, vol. 4, no. 3, pp. 178-187, 1978.

[CU95] W. Chen and H. Ural, "Synchronizable Checking Sequences Based on Multiple UIO Sequences," *IEEE/ACM Transactions on Networking*, vol. 3, pp. 152-157, 1995.

[FB91] S. Fujiwara, G. v. Bochmann, F. Khendek, M. Amalou, A. Ghedamsi, "Test selection based on finite state models," *IEEE Transactions on Software Engineering*, vol. 17 , no. 6, pp. 591-603, 1991.

[GG70] G. Gonenc, "A Method for the Design of Fault Detection Experiments," *IEEE Transactions on Computers*, vol. 19, no. 6, pp. 551-558, June 1970.

[GU95] S. Guyot and H. Ural, "Synchronizable Checking Sequences Based on UIO Sequences," *IFIP IWPTS'95*, Evry, France, pp. 395-407, Sept. 1995.

[HF64] F.C. Hennie, "Fault Detecting Experiments for Sequential Circuits," *Proceedings of the Fifth Annual Symposium on Switching Circuit Theory and Logical Design*, Princeton, N.J., pp. 95-110, 1964.

[HU02a] R.M. Hierons and H. Ural, "Reduced Length Checking Sequences," *IEEE Transactions on Computers*, vol. 51, no. 9, pp. 1111-1117, 2002.

[HU02b] R.M. Hierons and H. Ural, "UIO Sequence Based Checking Sequences for Distributed Test Architectures," Accepted for publication in *JIST*.

[II95] CAN/CSA-ISO/IEC Information technology – Opens Systems Interconnection – Conformance testing methodology and framework, 9646-1, Part 1: General Concepts, 1995.

[IS94] ISO/IEC JTC1/SC21/WG1, Revised Working Draft on Formal Methods in Conformance Testing, 1994.

[IS95] ISO/IEC Open Distributed Processing, Reference Model, 10748, Parts 1-4, 1995.

[KZ78] Z. Kohavi, *Switching and Finite Automata Theory*, McGraw-Hill, Inc.: New York, N.Y.

[LB94] G. Luo, R. Dssouli, G. v. Bochmann, P. Venkataram and A. Ghedamsi, "Test generation with respect to distributed interfaces," *Computer Standards and Interfaces*, vol. 16, pp. 119-132, 1994.

[LY94] D. Lee and M. Yannakakis, "Testing Finite State Machines: State Identification and Verification," *IEEE Transactions on Computers*, vol. 43, pp. 306-320, 1994.

[SB84] B. Sarikaya and G. v. Bochmann, "Synchronization and Specification Issues in Protocol Testing," *IEEE Transactions on Communications*, vol. 32, pp. 389-395, Apr. 1984.

[SD88] K.K. Sabnani and A.T. Dahbura, "A protocol test generation procedure," *Computer networks and ISDN systems*, vol. 15, no. 4, pp. 285-297, 1988.

[TY98] K.C. Tai and Y.C. Young, "Synchronizable Test Sequences of Finite State Machines," *Computer networks and ISDN systems*, vol. 13, pp. 1111-1134, Jul. 1998.

[UW93] H. Ural and Z. Wang, "Synchronizable test sequence generation using UIO sequences," *Computer Communications*, vol.16, pp. 653-661, 1993.

[UW97] H.Ural, X. Wu and F. Zhang, "On Minimizing the Lengths of Checking Sequences," *IEEE Transactions on Computers*, vol. 46, no. 1, pp. 93-99, January 1997.

[WD01] D. Whittier, "Solutions to Controllability and Observability Problems in Distributed Testing," Master's thesis, University of Ottawa, Canada, April 2001.

[YT98] Y.C. Young and K.C. Tai, "Observation Inaccuracy in Conformance Testing with Multiple Testers," *1st IEEE Workshop on Application - Specific Software Engineering and Technology*, pp. 80-85, March 1998.