Testing deterministic implementations from nondeterministic FSM specifications

A. Petrenko [1], *N. Yevtushenko* [2], *and G. v. Bochmann* [3]

1 *CRIM, Centre de Recherche Informatique de Montréal,*
1801 Avenue McGill College, Montréal, H3A 2N4, Canada,
Phone: (514) 398-3054, Fax: (514) 398-1244
petrenko@crim.ca

2 *Tomsk State University, 36 Lenin str., Tomsk, 634050, Russia,*
yevtushenko@elephot.tsu.tomsk.su

3 *Université de Montréal,*
CP. 6128, Succ. Centre-Ville, Montréal, H3C 3J7, Canada,
Phone: (514) 343-7535, Fax: (514) 343-5834,
bochmann@iro.umontreal.ca

Abstract

In this paper, conformance testing of protocols specified as nondeterministic finite state machines is considered. Protocol implementations are assumed to be deterministic. In this testing scenario, the conformance relation becomes a preorder, so-called reduction relation between FSMs. The reduction relation requires that an implementation machine produces a (sub)set of output sequences that can be produced by its specification machine in response to every input sequence. A method for deriving tests with respect to the reduction relation with full fault coverage for deterministic implementations is proposed based on certain properties of the product of specification and implementation machines.

Keywords

Conformance testing, test derivation, fault detection, I/O nondeterministic FSMs, equivalence and reduction relations

1 INTRODUCTION

Conformance testing of protocol implementations is often formalized as the FSM equivalence problem (Moore, 1956) and (Hennie, 1964). In particular, we are given two machines defined over the same input alphabet, one is referred to as the specification machine, the other is referred to as the implementation machine. The latter is treated as a black-box, so little is usually known about the implementation machine prior to testing; yet one typically assumes an upper bound on the number of its states (Gill, 1962). It is required to determine by testing whether the two are equivalent. A corresponding test suite is said to be complete with respect to equivalence relation in the class of implementation machines within the assumed bound on the number of states. The problem of deriving such a test suite for a given deterministic specification machine has recently attracted close attention in the literature (Vasilevski, 1973), (Chow, 1978), (Sidhu and Leung, 1989), (Fujiwara, Bochmann et al. 1991), (Ural, 1992), (Bochmann and Petrenko, 1994), (Yannakakis and Lee, 1995), and (Petrenko and Bochmann, 1996). Here we take a step further addressing a more general problem of testing a so-called reduction relation between FSMs (Petrenko, Bochmann, and Dssouli, 1993) and (Petrenko, Yevtushenko, and Bochmann, 1994). Specifically, we assume that an implementation machine is deterministic, but its specification machine is not necessarily deterministic. In this case, the implementation to be conforming is required to satisfy the reduction relation, i.e. the inclusion of output traces must hold for every input trace. The classical FSM equivalence problem becomes then a special case of the FSM reduction problem. A nondeterministic machine is evidently a more versatile paradigm for describing the protocol behavior than a deterministic one. A nondeterministic machine can represent, for example, a 'loose' description of the required behavior which contains options left for the protocol implementation. Most existing protocols allow these options. The nondeterministic machine paradigm is also useful for embedded testing. As shown in (Petrenko, Yevtushenko, and Dssouli, 1994) and (Petrenko, Yevtushenko, Bochmann, and Dssouli, 1996), testing a deterministic FSM embedded within a given system of communicating FSMs can be reduced to that of an appropriate nondeterministic FSM. Thus, the test derivation problem for the reduction relation is of both, theoretical and practical interests.

Not much work, however, has been done to solve this problem. In (Petrenko, Yevtushenko, Lebedev, and Das, 1993), it is demonstrated that the problem can be solved at least for a narrow subclass of nondeterministic FSMs. In this paper, we present a refined method for test derivation based on that work and analysis of properties of the product of the specification and implementation machines. The method is now extended to cover more general FSMs. Called the 'State-Counting method' (as in the previous work), it handles an arbitrary observable FSM which can be deterministic or not, completely or partially specified, and guarantees complete fault coverage within a given class of deterministic implementations with respect to the reduction relation. We also undertake a more profound study on state distinguishability in the context of the reduction relation.

This paper is organized as follows. Section 2 contains basic notions and definitions related to the model of a nondeterministic FSM. In Section 3, we present the SC-method for deriving test suites complete w.r.t. the reduction relation. The method is then extended in Section 4 to partially specified machines. In the concluding section, we discuss further research problems.

2 PRELIMINARIES

A finite state machine (FSM), often simply called a machine throughout this paper, is an initialized observable (possibly nondeterministic) Mealy machine which can be formally defined as follows. A *finite state machine* A is a 5-tuple (S, X, Y, h, s_0), where S is a finite set of n states with s_0 as the initial state; X - a finite set of input symbols; Y - a finite set of output symbols; and h - a behavior function $h: S\times X \rightarrow P(S\times Y)$, where $P(S\times Y)$ is the set of all nonempty subsets of $S\times Y$, such that $|\{s' \mid (s',y) \in h(s,x)\}| \leq 1$ for all $(s,x) \in S\times X$ and all $y \in Y$. (Starke, 1972). The machine A becomes deterministic when $|h(s,x)|=1$ for all $(s,x) \in S\times X$. In a deterministic FSM, instead of the behavior function which is required for expressing a nondeterministic behavior, we use two functions: the next state function δ, and the output function λ.

We extend the behavior function to a function on the set X^* of all input sequences containing the empty sequence ε, i.e., $h: S \times X^* \rightarrow P(S \times Y^*)$. For convenience, we use the same notation h for the extended function, as well, since in our discussions, this does not imply any contradiction. Assume $h(s,\varepsilon) = \{(s,\varepsilon)\}$ for all $s \in S$, and suppose that $h(s,\beta)$ is already specified. Then $h(s,\beta x) = \{ (s',\gamma y) \mid \exists\, s'' \in S\, [(s'',\gamma) \in h(s,\beta)\ \&\ (s',y) \in h(s'',x)] \}$.

The function h has two projections: the first projection h^1 and the second projection h^2, where $h^1(s,\alpha) = \{ s' \mid \exists\ \beta \in Y^*\, [(s',\beta) \in h(s,\alpha)] \}$ and $h^2(s,\alpha) = \{ \beta \mid \exists\ s' \in S\, [(s',\beta) \in h(s,\alpha)] \}$, for all $\alpha \in X^*$. Given $s \in S$, $\alpha \in X^*$, $\delta \in Y^*$, we use $h^1_\delta(s,\alpha)$ to denote a state (if exists) where the input sequence α takes the FSM A from s producing the output sequence δ.

Let $s \in S$, $\alpha \in X^*$ and $\delta \in h^2(s,\alpha)$. We say that the I/O sequence α/δ *visits* state s' from the state s if there exists a nonempty prefix β/γ of α/δ such that $h^1_\gamma(s,\beta) = s'$. In the case of a deterministic FSM A, we say that an input sequence α applied at the state s visits state s' of A if a non-empty prefix of β is a transfer sequence from the state s to state s'. Given states s, $s' \in S$, a sequence $\alpha \in X^*$ such that $h^1(s,\alpha) \ni s'$ is a *transfer* sequence from s to s'. If $h^1(s,\alpha) = \{s'\}$ then α is said to be a *deterministic transfer* sequence from s to s'. A state s is called *d-reachable* if there exits a deterministic transfer sequence from s_0 to s. For any state s, the empty sequence ε is a deterministic transfer sequence from s to s, therefore any FSM has at least one d-reachable state, namely the initial state. A set of (deterministic) transfer sequences from the initial state s_0 to all the (d-reachable) states of A is a (deterministic) *cover* of the FSM A. We consider here only connected machines. Given an FSM $A = (S, X, Y, h, s_0)$, A is said to be *connected* if for any state $s \in S$, there exists a transfer sequence $\alpha \in X^*$ from s_0 to s. In this paper, we use two types of covers for test derivation from an FSM. We use a traditional cover (often called a state cover set), defined as a set of transfer sequences used to take the machine A from the initial state to every its state. Note that, due to possible nondeterminism of the given machine, a single input sequence of a cover may serve as a transfer sequence for a number of states. We also use a deterministic cover V_A for the given FSM A. Obviously, in the class of deterministic machines, the two notions of a cover coincide.

To construct deterministic transfer sequences, we delete outputs from the FSM A and apply a standard technique for determinizing of the obtained automaton (Hopcroft and Ullman, 1979). A state s is a d-reachable state in A if and only if there exists a set $\{s\}$ among the states of the deterministic automaton. An input sequence that takes the automaton to the state $\{s\}$ is a deterministic transfer sequence from the initial state to the state s in the FSM A.

Given two states s of the FSM A and t of the FSM $B = (T, X, Y, H, t_0)$, state t is said to be a *reduction* of s, written $t \leq s$, if, for all input sequences $\alpha \in X^*$, the condition $H^2(t,\alpha) / h^2(s,\alpha)$ holds; otherwise t is not a reduction of s, written $t \not\leq s$. States s and t are *equivalent*, $s \cong t$, iff $s \leq t$ and $t \leq s$. On the class of deterministic machines, the relations coincide. We will also use weaker versions of equivalence and reduction relations, namely the E-equivalence and E-reduction, as well as their negations w.r.t. a given set E of input sequences, E/X^*; we use $\cong_E$, $\leq_E$, $\not\cong_E$ and $\not\leq_E$, to denote these relations, respectively. Given two machines, A and B, B is a reduction of A, written $B \leq A$, if the initial state of B is a reduction of the initial state of A. Similarly, the equivalence relation between machines is defined, $B \cong A$, iff $B \leq A$ and $A \leq B$.

Theorem 2.1. Given an FSM A, let B be a deterministic FSM over the same input alphabet. If $B \leq A$ then, for each state t of B, there exists a state s of A such that $t \leq s$ and, for each d-reachable state s of A, there exists a state t of B such that $t \leq s$.

Unlike to the case of deterministic FSMs, not every state of the nondeterministic FSM should correspond to some state of its reduction. However, as is established in Theorem 2.1, each d-reachable state of the FSM must have a corresponding state. Moreover, different states of the nondeterministic FSM may correspond to the same state of its reduction. The situation is similar to the case of compatible states of a partial deterministic FSM in context of state minimization (Grasselli and Luccio, 1965). We now establish necessary and sufficient conditions when two states of the FSM cannot correspond to a single state in any deterministic reduction of the FSM.

Given an FSM A, states s and r of A are said to be *r(1)-distinguishable* if there exists an input $x \in X$ such that $h^2(s,x) \cap h^2(r,x) = \varnothing$. Suppose we have determined all the pairs of r(j)-distinguishable states for $j=1,\ldots,k-1$; $k>1$. States s and r of A are said to be *r(k)-distinguishable* if they are r(j)-distinguishable, $j<k$, or there exists an input $x \in X$ such that for every output $y \in h^2(s,x) \cap h^2(r,x)$ states $h^1_y(s,x)$ and $h^1_y(r,x)$ are r(j)-distinguishable, $j<k$. Two states are said to be *r-distinguishable* if there exists an integer k such that the states are r(k)-distinguishable. Since the set S of states of A is finite there exists $k \le C^2_n$ such that the sets of pairs of r(k)-distinguishable and r(k+1)-distinguishable states coincide. By definition, any two separable states of the FSM A, i.e. states $s,r \in S$ for which there exists an input sequence $\alpha \in X^*$ such that $h^2(s,\alpha) \cap h^2(r,\alpha) = \varnothing$ (Starke, 1972), are r-distinguishable.

Theorem 2.2. Given an FSM A and states s and r of A, let B be a deterministic FSM over the same input alphabet as A. If there exists a state of B that is a reduction of the states s and r then the states s and r are not r-distinguishable.

Proof. Let $B = (T, X, Y, \Delta, \Lambda, t_0)$. 1. If there exists a state t of B, $t \le s$, $t \le r$, then the states s and r are not r(1)-distinguishable.

2. Assumption of induction. Let the statement of Theorem hold for all integers less than k, $k>1$, i.e. if a state of B is a reduction of the states s and r then the states s *and* r are not r(j)-distinguishable, for each j, $j<k$.

3. Suppose now that the states s and r are r(k)-distinguishable and there exists a state t of B, $t \le s$, $t \le r$. Then there exists an input $x \in X$ such that for every output $y \in h^2(s,x) \cap h^2(r,x)$ states $h^1_y(s,x)$ and $h^1_y(r,x)$ are r(j)-distinguishable, $j<k$. If t is a reduction of s and r then the state $t' = \Delta(t,x)$ of B should be a reduction of the states $h^1_y(s,x)$ and $h^1_y(r,x)$, where $y = \Lambda(t,x) \in h^2(s,x) \cap h^2(r,x)$. The latter contradicts the assumption of induction. Thus, if t is a reduction of states s and r then they are not r(k)-distinguishable for any k, i.e. they are not r-distinguishable.

❑

Combining Theorem 2.2 with the results of (Damiani, 1994), we have the following fact. A state of a deterministic FSM is not a reduction of two states of the FSM A if and only if these states are r-distinguishable.

The definition of r-distinguishable states implies an inductive procedure for constructing a set of input sequences r-distinguishing two given states s and r of the FSM A. We use $W(s,r)$ to denote this set. For any deterministic FSM B over the same input alphabet as A and any state t of B, the state t is not a reduction of both states s and r w.r.t. the set $W(s,r)$. The procedure for constructing $W(s,r)$ resembles that for determining the compatibility of states in a partial deterministic FSM (Grasselli and Luccio, 1965). We omit details, due to space constraints.

Based on the determined sets $W(s,r)$ for all pairs of r-distinguishable states, we define a so-called 'r-identifier' of a state of the FSM A as a set of sequences that r-distinguish the given state from any other r-distinguishable state of A. The union of r-distinguishing sets $W(s_i,s_j)$ over all states s_j of the FSM A that are r-distinguishable with s_i is said to be a (*harmonized*) *r-identifier* W_i of state s_i. The case $|W_i|=1$ resembles the notion of a UIO-sequence used in literature for deterministic FSMs. The set W_i becomes empty when state s_i cannot be r-distinguished from any other state. We define a *family* of harmonized r-identifiers as the set $\{W_i | s_i \in S\}$ and further call it simply a family of r-identifiers of the FSM A. The union of r-identifiers over all states of the FSM A is said to be an *r-characterization set* W of A. It is a generalization of a classical notion of a characterization set of a deterministic machine (Kohavi, 1978).

The equivalence and reduction relations serve the conformance relations between implementations and their FSM specifications for deriving test suites. Let a specification FSM A be defined over an input alphabet X. We assume that all potential implementations are represented by a set $\mathfrak{I}(X,Y')$ of deterministic FSMs defined over alphabets X and Y' (sometimes called a fault domain). A universal set of all deterministic FSMs with at most m states over input alphabet X is denoted by $\mathfrak{I}_m(X)$.

A test suite is a finite set E of finite input sequences of the FSM A. A test suite E is said to be *complete* for A w.r.t. the reduction relation in the class $\mathfrak{I}(X,Y')$ iff, for all $B \in \mathfrak{I}(X,Y')$, $B \not\leq A$ implies $B \not\leq_E A$. A test suite is said to be m-complete for A if it is complete in the fault domain $\mathfrak{I}_m(X)$.

Theorem 2.3. Given a specification FSM A over alphabets X and Y, a fault domain $\mathfrak{I}(X,Y)$ and a complete test suite E w.r.t. the reduction relation in the class $\mathfrak{I}(X,Y)$, let $\alpha\beta \in E$ be an input sequence where β is an sequence of length L, such that, for each output sequence γ of A to α, the set of output sequences of A to β at the state $h^1_\gamma(s_0,\alpha)$ contains each sequence of Y^* of length L. The complete test suite E reduced by replacing the sequence $\alpha\beta$ with α is complete in $\mathfrak{I}(X,Y)$.

Thus, in the case where implementations are known to preserve the output alphabet of their specification, a test suite can be reduced. We will, however, consider a more general case where the fault domain is the set $\mathfrak{I}_m(X)$.

3 CHECKING THE REDUCTION RELATION

In this section, we give a refined version of the method for test derivation based on an early version outlined in (Petrenko, Yevtushenko, Lebedev, and Das, 1993) for a rather narrow subclass of FSMs where each state is a d-reachable and the relation of r-distinguishabilility only includes pairs of separable states. The method is now extended to cover FSMs with states which are not d-reachable. We preserve the name 'State-Counting method' (SC-method for short); the name reflects the fact that test derivation for reduction relation relies upon counting appropriate states rather than upon checking individual transitions in conventional methods for equivalence relation. Another new feature of the SC-method is that state identification is now based on a more subtle distinguishability of states which may be nonseparable. We think to have also found a more appropriate technique for presenting the main ideas of the method which helps us find new avenues for further optimizing tests with guaranteed fault coverage. Our technique is based upon properties of the product of given specification and implementation machines.

3.1 Product of FSMs

Let A be a given (possibly nondeterministic) specification FSM and B be a deterministic implementation FSM over the input alphabet of A. Suppose that the FSM B is known. Then, to verify whether or not the FSM B is a reduction of A, we construct the product $A \times B$. Its initial state is the pair of initial states of the two machines A and B, the remaining states are determined by performing a reachability analysis. For a conforming implementation machine B that is a reduction of A, the output of the FSM B belongs to a set of outputs of A for any reachable state of the product and any input. The two machines, B and $A \times B$ are equivalent. If, however, B is not a reduction of A then there exists a reachable state of $A \times B$ and an input x such that the output of B is not in the set of outputs of A. In this situation, the machines cannot agree on any common output and the product is said to produce a special output *'fail'*. If the product at state (s,t) produces the output *'fail'* to an input x then we assume that the input x takes $A \times B$ from the state (s,t) to a designated state '*Fail*', called the *fail-state*. A sequence distinguishing B from A is a transfer sequence taking the product $A \times B$ from the initial state to the fail-state.

Formally, we define a product as follows. Let $A = (S, X, Y, h, s_0)$ be a given (possibly nondeterministic) specification FSM and $B = (T, X, Y', \Delta, \Lambda, t_0)$ be a deterministic implementation FSM. We define a machine $(S \times T \cup \{Fail\}, X, Y \cup \{fail\}, \psi, \varphi, s_0t_0)$, where for all $(s,t) \in S \times T$, $x \in X$,

$\psi(st,x) = [h^1_{\Lambda(s,x)}(s,x), \Delta(s,x)]$ and $\varphi(st,x) = \Lambda(t,x)$ if $h^2(s,x) \ni \Lambda(t,x)$;

otherwise $\psi(st,x) = Fail$ and $\varphi(st,x) = fail$.

$\psi(Fail,x) = Fail$ and $\varphi(Fail, x) = fail$, for all $x \in X$.

We use Q to denote the set of all states of this machine reachable from the initial state. Then $(Q, X, Y \cup \{fail\}, \psi, \varphi, q_0)$, where $q_0 = s_0t_0$, is called the *product* $A \times B$.

Assume now that we are required to test an unknown implementation FSM B against a given specification FSM A. We only know that the FSM B belongs to a given fault domain. Suppose that we could enumerate all machines in a given fault domain. Then a test suite for the FSM A complete in the fault domain could be obtained in a straightforward manner. In particular, for each FSM B, we construct the product $A \times B$ and determine at least one input sequence that takes the product from the initial state to the fail-state, whenever B is a nonconforming implementation machine. The union of such sequences for all machines in the fault domain gives a desired test suite. Such a solution can be costly, moreover, all the machines of the fault domain are simply not possible to enumerate in a realistic situation. There is a need for another approach that does not require each possible implementation machine separately.

The idea behind such an approach is based on the existence of certain properties shared by all input sequences causing, at least once, the output *'fail'* in the product $A \times B$ for each non-conforming FSM B of the given fault domain. As we shall show, based on these properties, a complete test suite can be derived without explicitly enumerating machines of a fault domain.

3.2 *M*-complete cover of an FSM

Given an FSM $A = (S, X, Y, h, s_0)$, a set of input sequences of A is said to be an *m-complete cover* for the FSM A if it is a cover of the product $A \times B$ for any FSM $B \in \Im_m(X)$. We use C_m to denote an m-complete cover for A.

Lemma 3.1. Given an FSM A and an m-complete cover C_m for A, the set C_m is an m-complete test suite for A.

Given the FSM A with n states and any $B \in \Im_m(X)$, the number of states in the product $A \times B$ does not exceed $mn+1$ and any state of this machine is reachable from its initial state by an input sequence whose length does not exceed mn. Thus, the set X^{mn} of all input sequences of length up to mn, is an m-complete cover for the FSM A with n states and, according to Lemma

3.1, an m-complete test suite for the FSM A (Petrenko, Yevtushenko, Lebedev, and Das, 1993).

Given a set of states $P \subseteq Q$ of the product $A \times B$ and state q', a sequence α is a *transfer sequence from P to q'*, if there exists a state $q \in P$ such that α is a transfer sequence from q to q'. If the length of the transfer sequence α from P to q' does not exceed that of any other sequence from P to q' then α is said to be a *minimal* transfer sequence from P to q'.

Let β be the set of all d-reachable states of A and V_A be a deterministic cover of the FSM A such that $|V_A|=|\beta|$. We use $P(V_A)$ to denote the set of states where the sequences of V_A take the product $A \times B$ from the initial state q_0. The set $P(V_A)$ contains $|\beta|$ states. Let also $\alpha_i \in V_A$ denote a deterministic transfer sequence of the FSM A from the initial state to a d-reachable state s_i. Since the product has at most $mn+1$ states, length of a minimal transfer sequence from the set $P(V_A)$ to any reachable state of the product does not exceed $mn-|P(V_A)|+1 = mn-|\beta|+1$. Therefore, the union of the sets $\alpha_i X^{mn-|\beta|+1}$ over all sequences $\alpha_i \in V_A$ is a cover of the product machine $A \times B$. It is also an m-complete cover for the FSM A, since B is an arbitrary FSM of the set $\Im_m(X)$.

Theorem 3.2. Given an FSM A, let V_A be a deterministic cover, and β be the set of all d-reachable states of A. Then the set $V_A X^{mn-|\beta|+1}$ is an m-complete test suite for the FSM A.

A test suite of Theorem 3.2 can often be reduced by deleting certain sequences from the set $X^{mn-|\beta|+1}$. Let $B \in \Im_m(X)$. Given an input sequence β, if among the states, visited by the transfer sequence β from a certain state of the set $P(V_A)$ to the fail-state of the product, either a state of the set $P(V_A)$ occurs or one state appears more than once, then the sequence β is not a minimal transfer sequence from $P(V_A)$ to the fail-state and can therefore be reduced. Based on this property of minimal transfer sequences from the set $P(V_A)$, we can construct an m-complete test suite for A as follows.

For any d-reachable state $s_j \in \beta$, we determine the traversal set $C_m(s_j)$ of input sequences as follows. An input sequence β is included into the set $C_m(s_j)$ if, for each sequence $\gamma \in h^2(s_j,\beta)$, there exists a d-reachable state $s \in \beta$ visited by β/γ exactly m times from the state s_j or there exists a state $s \in S \backslash \beta$ visited by β/γ exactly $m+1$ times while, for any proper prefix β' of β, there exists $\gamma' \in h^2(s_j,\beta')$ such that the property does not hold for the sequence β'/γ'.

We use $\alpha_j C_m(s_j)$ to denote the result of concatenation of a sequence $\alpha_j \in V_A$ that takes the FSM A from the initial state to the state $s_j \in \beta$, with all sequences of the set $C_m(s_j)$.

Theorem 3.3. Given an FSM A, a deterministic cover V_A of A, the union E of sets $\alpha_j C_m(s_j)$ over all $\alpha_j \in V_A$ is an m-complete test suite for A.

Proof. Let $B \in \Im_m(X)$ be a deterministic FSM and $P(V_A)$ be a set of states where the sequences from the set V_A take the product $A \times B$ from the initial state. If the state $Fail \in P(V_A)$ then an appropriate sequence $\alpha \in V_A$ is a transfer sequence from the initial state to the state $Fail$, and by construction, the state $Fail$ is visited by an appropriate sequence of the set E. Assume then that $Fail \notin P(V_A)$. Let an input sequence βx applied at some state $(s_j,t_j) \in P(V_A)$ be a minimal transfer sequence from $P(V_A)$ to $Fail$, i.e. the sequence $\alpha_j \beta x$ takes the product $A \times B$ from the initial state to the fail-state and γ be the output sequence of B to β at the state t_j. Since βx is a minimal transfer sequence from $P(V_A)$ to the state $Fail$, the pair β/γ is an I/O sequence of A at the state s_j. Moreover, the sequence β applied at the state $(s_j,t_j)\in P(V_A)$ is a minimal transfer sequence from $P(V_A)$ to the state q of $A\times B$, where β takes the product machine from the state (s_j,t_j).

Suppose that the sequence β/γ applied at the state s_j visits l times a state $s\in\beta$ of the FSM A and $l\geq m$. In this case, the sequence β applied at the state (s_j,t_j) visits l states $(s,t_1),\ldots,(s,t_l)$ of the product $A\times B$. Since the FSM B has at most m states and the set $P(V_A)$ contains a pair (s,t) for an appropriate state t of the FSM B, among these states either a state from the set $P(V_A)$ occurs or at least one state appears more than once. In the both cases, the sequence β is not a minimal transfer sequence from $P(V_A)$ to q.

Similar to this, β is not a minimal transfer sequence from $P(V_A)$ to q if β visits l times a state $s\in S\backslash\beta$ of the FSM A and $l\geq m+1$. Thus, the sequence β is a proper prefix of an appropriate sequence of the set $C_m(s_j)$ and there exists a sequence $\alpha_j\alpha\in E$ with a prefix $\alpha_j\beta x$ that visits the state $Fail$ from the initial state of the product $A\times B$.

❑

Compared to Theorem 3.2, Theorem 3.3 offers a more economical way of constructing an m-complete test suite. To assure that a set of input sequences of the specification FSM A visits the state $Fail$ of the product $A\times B$ for any $B\in\Im_m(X)$ we include in the traversal set $C_m(s_j)$ each input sequence if there may exist a product machine $A\times B$, $B\in\Im_m(X)$, such that the sequence may turn to be a minimal transfer sequence from $P(V_A)$ to $Fail$. A sequence $\beta\in C_m(s_j)$ is expanded by appending all inputs until it visits an appropriate state s of the FSM A m or $m+1$ times for each output sequence of A to β at the state s_j. The size of the traversal sets $C_m(s_j)$ is exponential and it is, therefore, important to determine cases where their sequences can be terminated as early as possible. For a specification FSM such that none of its states are r-distinguishable and no state is a reduction of any other state, it seems nearly impossible to reduce the size of the traversal sets. Certain sufficient conditions enforcing an earlier termination of sequences of the traversal sets $C_m(s_j)$ can be established provided that a given specification FSM A has r-distinguishable states.

3.3 Reducing traversal sets

Let an FSM A have states, say, s_1 and s_2, r-distinguished by a set $W(s_1,s_2)$ of input sequences. Given an FSM B, let the product $A\times B$ have states (s_1,t) and (s_2,t) for an appropriate state t of the FSM B. Then we refer to these states as to *conflicting* states. Due to the properties of the set $W(s_1,s_2)$, there exists an input sequence $\alpha\in W(s_1,s_2)$ such that the output response of B to the input sequence α at the state t is not in the set of output sequences of the FSM A to α at least at one of states s_1 and s_2. Thus, the input sequence α takes the product $A\times B$ at least from one of the states (s_1,t) and (s_2,t) to the fail-state. In other words, if a transfer sequence β applied at some state of the product visits the two conflicting states, the sequence α applied at (s_1,t) or (s_2,t) could be used as a shortcut to reach the fail-state in this product.

We now analyze a string of states visited by a minimal transfer sequence β from the set of states $P(V_A)$ to the fail-state of the product $A\times B$ and establish sufficient conditions when the set of states visited by the β, along with states of the set $P(V_A)$, contains conflicting states.

Lemma 3.4. Given FSMs A and B, $B\in\Im_m(X)$, a set D of pairwise r-distinguishable states of A together with its subset ∂ of d-reachable states, let an input sequence $\beta\nu$, $\nu\neq\varepsilon$, applied at some state $(s,t) \in P(V_A)$ be a minimal transfer sequence from $P(V_A)$ to the fail-state of $A \times B$ and γ be the output response of B to the sequence β applied at the state t. If the I/O sequence β/γ, applied at the state s of A, visits states of D more than m-$|\partial|$ times then the set of states visited by the β, applied at the state (s,t) of $A \times B$, together with the states of $P(V_A)$, contains conflicting states (s_1,t') and (s_2,t'), $s_1,s_2 \in D$.

Proof. Let $B=(T, X, Y, \Delta, \Lambda, t_0) \in \Im_m(X)$, and the sequence $\beta\nu$ applied at a state $(s,t) \in P(V_A)$ be a minimal transfer sequence from $P(V_A)$ to the fail-state of $A \times B$ and $\gamma = \Lambda(t,\beta)$. Then the pair β/γ is an I/O sequence of the FSM A. Suppose that the I/O sequence β/γ applied at the state s of A visits l times states of D and $l>m$-$|\partial|$. In this case, the sequence β applied at the state (s,t) traverses l states $(s_1,t_1),...,(s_l,t_l)$ of the product, where $s_i \in D$, $i=1,\dots,l$. Since the FSM B has at most m states, among the states $(s_1,t_1),...,(s_{m-|\partial|+1},t_{m-|\partial|+1})$ combined with states of the set $\{(s',t')|\ s' \in \partial\} \subseteq P(V_A)$, there can be at most m states with pairwise distinct states of the FSM B. The set $P(V_A)$ contains at least $|\partial|$ distinct pairs (s',t'), $s' \in D$. Thus, at least two states in the union of the sets $\{(s_i,t_i)|\ i=1,...,m$-$|\partial|+1\}$ and $\{(s',t'),\ s' \in \partial\} \subseteq P(V_A)$ have the same state of B. Because of β being a minimal transfer sequence from $P(V_A)$ to the fail-state, the corresponding two states of the product cannot coincide. Thus, among the states (s_1,t_1), ..., $(s_{m-|\partial|+1},t_{m-|\partial|+1})$ visited by β and states of the set $P(V_A)$, there exist two distinct states (s_1,t') and (s_2,t'), $s_1,s_2 \in D$, for an appropriate state t' of the FSM B.

❑

Let β be an input sequence. We denote $\Im_m(\alpha_j\beta)$, where $\alpha_j \in V_A$, the set of all implementation FSMs $B \in \Im_m(X)$ such that an input sequence with the prefix β applied at the state $(s_j,t_j) \in P(V_A)$ is a minimal transfer sequence from $P(V_A)$ to the fail-state of the product $A \times B$. Based on Lemma 3.4, the following statement can be established.

Lemma 3.5. Given FSM A, an input sequence β, an r-characterization set W and a d-reachable state s_j of A, let for each $\gamma \in h^2(s_j,\beta)$, there exists a set D of pairwise r-distinguishable states of A such that the I/O sequence β/γ, applied at the state s_j of A, visits states of D more than m-$|\partial|$ times, where ∂ is the subset of d-reachable states of D. Then the union of the sets $\alpha_i W$ over all $\alpha_i \in V_A$, and the sets $\alpha_j\beta' W$ over all nonempty prefixes β' of β is a test suite complete for A in the class $\Im_m(\alpha_j\beta)$.

The lemma states that replacing an exponential expansion (Theorem 3.3.) of an input sequence for which the conditions of Lemma 3.5 hold, by a certain polynomial set of input sequences preserves the fault coverage. Thus, an m-complete test suite can now be obtained as a union of corresponding sets over all d-reachable states of A and input sequences satisfying Lemma 3.5. However, a test suite complete in the class $\Im_m(\alpha_j\beta)$ can often be reduced if we use a family of state r-identifiers instead of an r-characterization set W. The procedure for deriving a test suite $T_m(\alpha_j\beta)$ complete for A in the class $\Im_m(\alpha_j\beta)$ for the sequence β satisfying Lemma 3.5, includes the following steps.

1. Find a deterministic cover V_A of the FSM A.

2. For each $\alpha_i \in V_A$, that takes the FSM A from the initial state to a d-reachable state s_i, concatenate α_i with every sequence of the set W_i. Let E be the union of obtained sets over all $\alpha_i \in V_A$.
3. For each nonempty prefix β' of the sequence β determine $h^1(s_0,\alpha_j\beta')$. Then concatenate $\alpha_j\beta'$ with all sequences of every W_i, $s_i \in h^1(s_j,\alpha_j\beta')$. Let $\alpha_j(\beta @ \{W_i | s_i \in S\})$ denote the result of this concatenation.
4. Find the union $T_m(\alpha_j\beta)$ of E and $\alpha_j(\beta @ \{W_i | s_i \in S\}$.

Theorem 3.6. Given an FSM A, let $T_m(\alpha_j\beta)$ be the set of input sequences derived from A by the above procedure. Then the set $T_m(\alpha_j\beta)$ is a complete test suite for the FSM A in the class $\Im_m(\alpha_j\beta)$.

Proof. Let $B=(T, X, Y, \Delta, \Lambda, t_0) \in \Im_m(\alpha_j\beta)$, the sequence $\beta\nu$ applied at a state $(s_j,t_j) \in P(V_A)$ be a minimal transfer sequence from $P(V_A)$ to the fail-state of $A \times B$ and $\gamma = \Lambda(t,\beta)$. If $\nu = \varepsilon$ then $Fail \in P(V_A)$. If the state $Fail \in P(V_A)$ then an appropriate sequence $\alpha \in V_A$ is a transfer sequence from the initial state to the state $Fail$. Let then $\nu \neq \varepsilon$. Then $\beta\gamma$ is an I/O sequence of A. The sequence β satisfies the conditions of Lemma 3.5; therefore, there exists a set D of pairwise r-distinguishable states of A such that the I/O sequence β/γ, applied at the state s_j of A, visits states of D more than m-$|\partial|$ times. Due to Lemma 3.4, among the states visited by β applied at state (s_j,t_j) and states of the set $P(V_A)$, there exist two distinct states (s_1,t) and (s_2,t), $s_1,s_2 \in D$, for an appropriate state t of the FSM B. Thus, among sequences V_A and sequences $\alpha_j\beta'$, where β' is a nonempty prefix of β, there exist sequences α' and α'' that take the product to the states (s_1,t) and (s_2,t), where $s_1,s_2 \in D$. The states s_1 and s_2 of A are r-distinguished by an appropriate sequence $\delta \in W_1 \cap W_2$ and, by construction, $\alpha'\delta, \alpha''\delta \in T_m(\alpha_j\beta)$. Thus, at least one of the two sequences, $\alpha'\delta$ or $\alpha''\delta$, takes the product to the fail-state.

❑

3.4 The SC-Method

Based on Theorem 3.6, an m-complete test suite can now be derived as the union of test suites complete in classes $\Im_m(\alpha_j\beta)$ over all sequences $\alpha_j \in V_A$ and all input sequences β such that, for d-reachable state s_j of A and each sequence $\gamma \in h^2(s_j,\beta)$, there exists a set D of pairwise r-distinguishable states of A such that the I/O sequence β/γ, applied at the state s_j of A, visits states of D more than m-$|\partial|$ times. The SC-method for constructing an m-complete test suite includes the following steps.

1. Find a deterministic cover V_A of the FSM A.
2. Find all pairs of r-distinguishable states of A and determine all maximal sets $D_1,...,D_k$ of pairwise r-distinguishable states. For each D_r, $r=1,...,k$, find a maximal subset ∂_r of d-reachable states.
3. Construct a family $\{W_i | s_i \in S\}$ of r-identifiers of A.

4. For any d-reachable state s_j, derive the traversal set $Tr_m(s_j)$ as follows. An input sequence β is included into the set $Tr_m(s_j)$ if, for each sequence $\gamma \in h^2(s_j,\beta)$, there exists a set D_r such that its states are visited by β/γ exactly $(m-|\partial_r|+1)$ times from the state s_j.
5. For each traversal set $Tr_m(s_j)$ and every sequence $\beta \in Tr_m(s_j)$, construct the test suite $T_m(\alpha_j\beta)$ complete in the class $\Im_m(\alpha_j\beta)$, by use of the above given procedure (Section3.3).
6. Find the union E of $T_m(\alpha_j\beta)$ for all $\alpha_j \in V_A$ and $\beta \in Tr_m(s_j)$ (note that each sequence that is a prefix of another sequence can be deleted from E to simplify the result).

Theorem 3.7. Given an FSM A, let E be the set of input sequences derived from A by the SC-method. Then the set E is an m-complete test suite for the FSM A.

Proof. Consider an m-complete test suite $V_A X^{mn-|\beta|+1}$ from Lemma 3.2. Let $B \in \Im_m(X)$, and the sequence ν, $\nu \in X^{mn-|\beta|+1}$, applied at some state (s_j,t_j) be a minimal transfer sequence from $P(V_A)$ to the fail-state of $A \times B$. Determine a minimal prefix β of ν such that $\beta \in Tr_m(s_j)$. Due to Theorem 3.6, a test suite $T_m(\alpha_j\beta) \subseteq E$ contains an input sequence that takes the product $A \times B$ to the fail-state.

❑

Example. We consider the FSM A shown in Figure 1. State 3 cannot be deterministically reached from the initial state 1, all the other states are d-reachable. We choose a minimal d-reachable state cover set $V_A=\{\varepsilon, a, ab\}$, the empty sequence ε serves a transfer sequence for the initial state, a for state 2, and ab for state 4. Next, we check whether the states are r-distinguishable. The sequence a r-distinguishes states 2 and 3; the sequence aa r-distinguishes states 1 and 2; aaa - states 1 and 3. States 2 and 4 are r-distinguished by the sequence b; the states 3 and 4 - by the sequence bb. States 1 and 4 are not separable but they are r-distinguished by the set $\{aaa, ab\}$ of input sequences. In fact, there are two common output responses x and y of A to the input a at the states 1 and 4. The I/O sequence a/x takes the FSM A from the states 1 and 4 to the states 2 and 4 which are separated by the input sequence b while the I/O sequence a/y takes the FSM A from the states 1 and 4 to the states 2 and 1, respectively, which are r-distinguished by the input sequence aa.

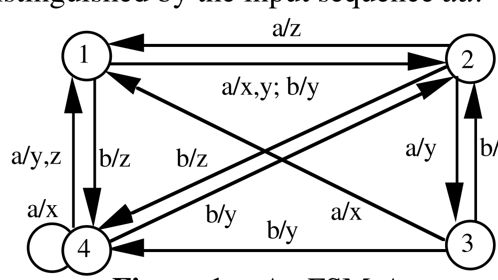

Figure 1 An FSM A.

States 1,2, 3, and 4 form a single maximal set D of pairwise r-distinguishable states and $W_1 = \{aaa, ab\}$, $W_2 = \{aa, b\}$, $W_3 = \{a, bb\}$, and $W_4 = \{aaa, ab, bb\}$. State 3 is not a d-reachable state, so the subset of pairwise r-distinguishable d-reachable states is $\partial = \{1,2,4\}$. We assume that the number of states in any implementation is at most four (m=4) and proceed by determining traversal sets for d-reachable states. The termination rule for expanding input sequences becomes $m-|\partial|+1 = 4-3+1=2$, in other words, states from D should be visited twice before any input sequence can be terminated. Since D contains all the states of the FSM A, it is required to apply X^2 at each d-reachable state, thus $Tr_4(i)=\{a,b\}^2$, for each i=1,2,4. The union of complete test suites $T_4(\alpha_i\beta)$ over all sequences $\alpha_j\beta$, $\alpha_j \in V_A$, $\beta \in Tr_4(j)$ is an m-complete test suite (m=4). As an example, consider the sequence $aab \in a\{a,b\}^2$. One can assure that $h^1(1,a) = \{2\}$, and so the sequence a should be concatenated with W_2; $h^1(1,aa) = \{1,3\}$ and the sequence aa is concatenated with $W_1 \cup W_3$; $h^1(1,aab) = \{1,2\}$, thus, the sequence aab is concatenated with $W_1 \cup W_2$.

4 EXTENSION TO PARTIALLY SPECIFIED MACHINES

The model of partially specified finite state machines is useful for describing the behavior of systems where transitions out of certain states on some inputs are not defined, these are 'don't care' transitions. Implementation machines are usually assumed to be completely specified. Implementing a partial specification amounts to completing it in a certain way. The model defined in Section 2 is, in fact, a completely specified (complete) finite state machine. Now we formally define a partial FSM (PFSM) and generalize the reduction and equivalence relations.

A *partial* finite state machine A is an observable partial FSM, i.e. 6-tuple (S, X, Y, h, s_0, D_A), where S is a set of states with s_0 as the initial state; X - a finite set of input symbols; Y - a finite set of output symbols; D_A - a specification domain, $D_A \subset S \times X$; and h - a behavior function $h\colon D_A \to P(S \times Y)$ such that $|\{s' \mid (s',y) \in h(s,x)\}| \le 1$ for all $(s,x) \in D_A$ and all $y \in Y$. Replacing D_A by $S \times X$, we obtain a complete FSM.

Any I/O sequence specified in an observable machine takes it from its initial state to a unique state. However, in a nondeterministic machine, a specified input sequence may lead to several states. Generally speaking, these states may have different unspecified inputs. Here, we restrict ourselves to a class of machines with so-called harmonized traces (Petrenko, Yevtushenko, Lebedev, and Das, 1993). States of such a machine once reached from the initial state with the same specified transfer sequence have the same set of specified (unspecified) inputs. Figure 2 shows an example of a partial machine with harmonized traces. Each input sequences specified at the initial state of such machine does not execute any 'don't care' transition. We use X_A^* to denote the set of all sequences specified at the initial state.

To test a machine against its PFSM specification, we have to compare the I/O behaviors of a complete implementation FSM $B = (T, X, Y, H, t_0)$ and a partial specification FSM A (S, X, Y, h, s_0, D_A).

An FSM B is a *quasi-reduction* of a PFSM A, written $B \le_{\text{quasi}} A$, iff for all input sequences $\alpha \in X_A^*$ the condition $H^2(t_0,\alpha) \subseteq h^2(s_0,\alpha)$ holds; otherwise $B \not\le_{\text{quasi}} A$.

An FSM B is *quasi-equivalent* to a PFSM A, written $B \cong_{\text{quasi}} A$, iff for all input sequences $\alpha \in X_A^*$ the condition $H^2(t_0,\alpha) = h^2(s_0,\alpha)$ holds; otherwise $B \not\cong_{\text{quasi}} A$. This relation originates from the quasi-equivalence relation introduced in (Gill, 1962) for deterministic machines which corresponds to a so-called weak conformance (Sidhu and Leung, 1989), (Yannakakis and Lee, 1995). On the class of deterministic machines, quasi-reduction and quasi-equivalence coincide.

According to definitions of quasi-equivalence and quasi-reduction relations, deriving test suites, we should omit input sequences on which the behavior of the specification machine is not defined. Thus, all complete test suites can be determined as subsets of the set X_A^*. With this exception, the definitions of complete test suites for partial machines repeat that for complete machines.

A partial machine A with harmonized traces can often be treated as a special complete nondeterministic machine a by treating its transitions on unspecified inputs as 'don't care' transitions to a *trap* state (Unger, 1969). Such transitions are labeled with an input not specified at the current state of A and all outputs of some superset of Y. The superset Y' of Y

represents all outputs in the class of implementation machines. The trap state has looping transitions labeled with all inputs in X and all outputs. Input sequences leading a into the trap state are sequences not specified in A, they constitute the set $X^{*}\backslash X_A^{*}$. The machine a is said to be the *completed form* of A. The completed form of a PFSM reflects a rather general completeness assumption, namely 'undefined by default', used in protocol testing (Petrenko, Bochmann, and Dssouli, 1993). Figure 2 shows an example. Here 'any' stands for an arbitrary output in Y', a 'black hole' represents a trap state. The completed form is necessary nondeterministic even when a given machine is deterministic.

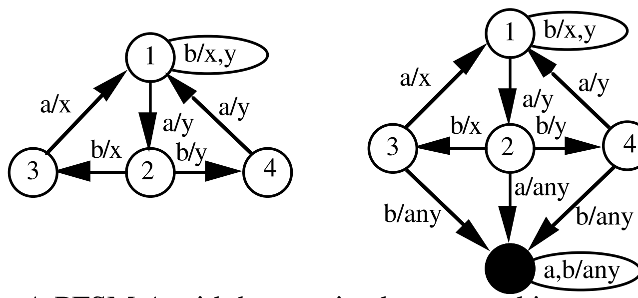

Figure 2 A PFSM A with harmonized traces and its completed form.

By construction, the completed form a of a PFSM A with harmonized traces is a machine that is quasi-equivalent to A. In fact, for every input sequence specified in A, the sets of output sequences produced by their initial states coincide, i.e. $a \cong_{\text{quasi}} A$. (It is not necessarily true for an arbitrary partial nondeterministic machine). Every deterministic machine has harmonized traces, therefore, the problem of test derivation from a partial deterministic FSM w.r.t. the quasi-equivalence relation and that for a complete nondeterministic FSM w.r.t. the reduction relation become equivalent problems. We have an even more general fact.

Theorem 4.1. Let a be the completed form of a partial machine A with harmonized traces. Assume that an FSM B over the same input and output alphabets is complete and deterministic. Then $B \leq_{\text{quasi}} A$ if and only if $B \leq a$.

Proof. I. $B \leq_{\text{quasi}} A \Rightarrow B \leq a$. Let $A = (S, X, Y, h, s_0, D_A)$ be a partial nondeterministic machine, $a = (S, X, Y, h, s_0, D_A)$ be its completed form, and $B = (T, X, Y, \delta, \lambda, t_0)$ be a complete deterministic machine. Assume that $B \leq_{\text{quasi}} A$, but $B \not\leq a$. In this case, there exists an input sequence α such that $\lambda(t_0,\alpha) \notin h^2(s_0,\alpha)$. By virtue of definition of the completed form, $a \cong_{\text{quasi}} A$, that is $h^2(s_0,\beta) = h^2(s_0,\beta)$ for all input sequences $\beta \in X_A^{*}$ Assuming $\alpha \in X_A^{*}$ leads us to a contradiction, as $\lambda(t_0,\alpha) \notin h^2(s_0,\alpha)$ and B is not a quasi-reduction of A. Suppose therefore that $\alpha \notin X_A^{*}$ and $\alpha = \beta\gamma$, where $\beta \in X_A^{*}$ and $\gamma \in X^{*}$. A has harmonized traces, then its completed form a in each state of the set $h^1(s_0,\alpha)$ produces in response to γ all output sequences of the length of γ. Then $\lambda(t_0,\beta) \in h^2(s_0, \beta)$, and $\lambda(\delta(t_0,\beta),\gamma) \in h^2(s,\gamma)$ for any $s \in h^1(s_0,\beta)$. This again leads us to a contradiction.

II. $B \leq a \Rightarrow B \leq_{\text{quasi}} A$. Assume that $B \leq a$, but $B \not\leq_{\text{quasi}} A$. If B is not a quasi-reduction of A then there exists an input sequence $\alpha \in X_A^{*}$ such that $\lambda(t_0,\alpha) \notin h^2(s_0,\alpha)$. $a \cong_{\text{quasi}} A$, it means that for all input sequences $\alpha \in X_A^{*}$ the condition $h^2(s_0,\alpha) = h^2(s_0,\alpha)$ holds. Thus, $\lambda(t_0,\alpha) \notin h^2(s_0,\alpha)$. A contradiction.

❑

Based on this theorem, the problem of test derivation from a partial FSM with harmonized traces w.r.t. the quasi-reduction relation can be reduced to that from its completed form w.r.t. reduction relation. The SC-method serves this purpose. It follows, however, from our discussions that the trap state does not require any identification (anyway, every other state is a reduction of the trap state) neither should transitions into the trap state be covered by a test suite. In other words, we have the following fact as a corollary to the above theorem.

Corollary. Let E be a complete test suite for the completed form a of a partial machine A with harmonized traces for the reduction relation in the class of deterministic implementation machines. Then $E \cap X_A^*$ is a complete test suite for A w.r.t. the quasi-reduction relation in the same class of implementations.

Note that constructing a complete test suite exclusively from the set X_A^* of specified input sequences becomes essential in situations where undefined transitions are treated as 'forbidden' transitions, as explained in (Yevtushenko and Petrenko, 1990), (Petrenko, 1991), (Petrenko and Yevtushenko, 1992), (Luo, Petrenko, and Bochmann, 1994), and (Yannakakis and Lee, 1995). The difference from the latter work is that we consider here a wider class of partial machines that are not necessarily reduced. (Yannakakis and Lee, 1995) gives no solution for partial machines with compatible, i.e. indistinguishable states, but our method is fully applicable to such machines.

5 CONCLUSION

We have presented a refined version of the test derivation method (SC-method) which, for a given FSM, generates a test suite in the context of the reduction relation. The SC-method is proven to provide full fault coverage on the pre-determined class of deterministic implementations. It can be applied to various classes of specification FSMs, including partially specified machines with compatible states provided that they are observable. This limitation is by no means prohibitive, as any FSM with harmonized traces has an equivalent observable form. Our method follows a new principle of constructing test sequences, namely counting appropriate states visited by test sequences, unlike conventional methods that strictly follow the transition checking principle.

Next step in this direction would be to further elaborate the proposed approach taking into account, for example, that the reduction relation may hold between a number of states in a given specification machine, all these states can correspond to a single state of an implementation FSM. It is also interesting to establish which states of a specification machine (along with d-reachable states) should have a corresponding state in an implementation FSM.

Acknowledgments

This research was partly supported by an NSERC strategic research grant and by the Russian Found for Fundamental Research. The authors would like to thank the anonymous referees for their comments that helped improve the presentation of this paper.

6 REFERENCES

Bochmann, v.G. and Petrenko, A. (1994) Protocol testing: review of methods and relevance for software testing. *ISSTA'94 ACM International Symposium on Software Testing and Analysis*. pp. 109-124

Chow, T. S. (1978) Testing software design modeled by finite-state machines. *IEEE Transactions on Software Engineering*, Vol. SE-4, No 3, pp. 178-187.

Damiani, M. (1994) Nondeterministic finite-state machines and sequential don't cares. *Proceedings of the European Conference on Design&Test,* pp. 192-198.

Fujiwara, S., Bochmann, v.G., Khendek, F., Amalou, M., and Ghedamsi, A. (1991) Test selection based on finite state models. *IEEE Transactions on Software Engineering,* Vol. SE-17, No. 6, pp. 591-603.

Gill, A. (1962) *Introduction to the theory of finite-state machines,* NY, McGraw-Hill, 270p.

Grasselli, A. and Luccio, F. (1965) A method for minimizing the number of internal states in incompletely specified sequential networks. *IEEE Transactions on Electronic Computers,* No. 6, pp. 350-359.

Hennie, F. C. (1964) Fault detecting experiments for sequential circuits. *Proceedings of the IEEE 5th Ann. Symp. on Switching Circuits Theory and Logical Design,* pp. 95-110.

Hopcroft, J.E., and Ullman, J.D. (1979) Introduction to automata theory. languages and computation. Addison-Wesley Publishing Company, Inc., 418 p.

Kohavi, Z. (1978) *Switching and finite automata theory,* N.Y., McGraw-Hill.

Luo, G., Petrenko, A., and Bochmann, v.G. (1994) Selecting test sequences for partially-specified nondeterministic finite state machines. *Protocol Test Systems VII (the Proceedings of IFIP WG 6.1 International Workshop on Protocol Test Systems 1994),* Chapman & Hall, 1995, pp. 95-110.

Moore, E. F. (1956) Gedanken-experiments on sequential machines, *Automata Studies,* Princeton University Press, Princeton, NJ, pp. 129-153.

Petrenko, A. (1991) Checking experiments with protocol machines. *IFIP Transactions, Protocol Test Systems, IV (the Proceedings of IFIP TC6 Fourth International Workshop on Protocol Test Systems, 1991),* 1992, North-Holland, pp. 83-94.

Petrenko, A. and Yevtushenko, N. (1992) Test suite generation for a fsm with a given type of implementation errors. *IFIP Transactions Protocol Specification, Testing, and Verification XII (the Proceedings of IFIP TC6 12th International Symposium on Protocol Specification, Testing, and Verification 1992)*, pp. 229-243.

Petrenko, A., Bochmann, v.G., and Dssouli, R. (1993) Conformance relations and test derivation. *IFIP Transactions Protocol Test Systems VI (the Proceedings of IFIP TC6 Fifth International Workshop on Protocol Test Systems,1993)* North-Holland, 1994, pp. 157-178.

Petrenko, A., Yevtushenko, N., Lebedev, A., and Das, A. (1993) Nondeterministic state machines in protocol conformance testing. *IFIP Transactions Protocol Test Systems VI (the Proceedings of IFIP TC6 Fifth International Workshop on Protocol Test Systems 1993),* North-Holland, 1994, pp. 363-378.

Petrenko, A., Yevtushenko, N., and Dssouli, R. (1994) Testing strategies for communicating fsms. *Protocol Test Systems VII (the Proceedings of IFIP WG 6.1 International Workshop on Protocol Test Systems, 1994)* Chapman & Hall, 1995, pp. 193-208.

Petrenko, A., Yevtushenko, N., and Bochmann, v.G. (1994) Experiments on nondeterministic systems for the reduction relation. *Technical Report 932, Université de Montréal*, 23p.

Petrenko, A. and Bochmann, v.G. (1996) On fault coverage of tests for finite state specifications. To appear in a special issue on Protocol Testing of *Computer Networks and ISDN Systems*.

Petrenko, A., Yevtushenko, N., Bochmann, v.G., and Dssouli, R. (1996) Testing in context: framework and test derivation. *Technical Report 1011, Université de Montréal,* To appear in a special issue on Protocol Engineering of *Computer Communications Journal*.

Starke, P. H. (1972) *Abstract automata.* North-Holland/American Elsevier, 419p.

Sidhu, D. P. and Leung, T. K. (1989) Formal methods for protocol testing: a detailed study. *IEEE Transactions on Software Engineering*, Vol SE-15, No 4, pp. 413-426.

Vasilevski, M. P. (1973) Failure diagnosis of automata. *Cybernetics*, Plenum Publishing Corporation, New York, No 4, pp. 653-665.

Unger, S. H. (1969) *Asynchronous sequential switching circuits* Wiley-Interscience.

Ural, H. (1992) Formal methods for test sequence generation. *Computer Communications,* Vol. 15, No. 5, pp. 311-325.

Yannakakis, M. and Lee, D. (1995) Testing finite state machines: fault detection. *Journal of Computer and System Sciences,* 50, pp. 209-227.

Yevtushenko, N. and Petrenko, A. (1989) Fault-detection capability of multiple experiments. *Automatic Control and Computer Sciences,* Allerton Press, Inc., N.Y., Vol. 23, No. 3, pp. 7-11.

Yevtushenko, N. and Petrenko, A. (1990) A method of constructing a test experiment for an arbitrary deterministic automaton. *Automatic Control and Computer Sciences,* Allerton Press, Inc., N.Y., Vol. 24, No. 5, pp. 65-68.

7 BIOGRAPHY

Alexandre Petrenko received the Diploma degree in electrical and computer engineering from Riga Polytechnic Institute in 1970 and the Ph.D. in computer science from the Institute of Electronics and Computer Science, Riga, USSR, in 1974. In 1996, he has joined CRIM, Centre de Recherche Informatique de Montréal, Canada. From 1992 to 1996, he was a visiting professor/researcher of the Université de Montréal. From 1982 to 1992, he was the head of a research department of the Institute of Electronics and Computer Science in Riga. From 1979 to 1982, he was with the Networking Task Force of the International Institute for Applied Systems Analysis (IIASA), Vienna, Austria. From 1969 to 1979, he was a researcher and the head of a research department of the Institute of Electronics and Computer Science in Riga. His current research interests include high-speed networks, communication software engineering, formal methods, conformance testing, and testability.

Nina Yevtushenko received the Diploma degree in radio-physics in 1971 and Ph. D. in computer science in 1983, both from the Tomsk State University, Russia. She is currently a Professor at that University. Her research interests include the automata and FSM theory and testing problems.

Gregor v. Bochmann received the Diploma degree in physics from the University of Munich, Munich, West Germany, in 1968 and the Ph.D. degree from McGill University, Montreal, P.Q., Canada, in 1971. He has worked in the areas of programming languages, compiler design, communication protocols, and software engineering and has published many papers in these areas. He holds the Hewlett-Packard-NSERC-CITI chair of industrial research on communication protocols in the Département d'informatique et de recherche opérationnelle, Université de Montréal. His present work is aimed at design methods for communication protocols and distributed systems. He has been actively involved in the standardization of formal description techniques for OSI. He is presently one of the scientific directors of the Centre de Recherche Informatique de Montréal (CRIM).