



Bounded sequence testing from deterministic finite state machines

Florentin Ipate*

Department of Computer Science, University of Pitesti, 110040 Pitesti, Romania

ARTICLE INFO

Article history:

Received 18 February 2008

Received in revised form 10 November 2009

Accepted 28 January 2010

Communicated by M. Ito

Keywords:

Finite state machines

Test selection

W-method

Wp-method

Specification-based testing

ABSTRACT

The *W*- and *Wp*-methods are the basis for conformance testing from a *deterministic finite state machine (DFSM)* when the conformance relation considered is equivalence. However, many DFSM applications use only input sequences of limited length. In such cases, the test data only need to establish that the implementation under test produces the specified responses for sequences of length less than or equal to the upper bound *l*. This paper extends the *W*- and *Wp*-methods to the case in which only *bounded sequences* are allowed. The methods for bounded sequences are stronger than the originals since test suites for the unbounded case can be obtained as a particular case (in which the upper bound *l* is sufficiently large) from the new formulae. Furthermore, the generalization is not straightforward as it is not sufficient to extract the sequences of length at most *l* from the test suites produced in the unbounded case, or even all prefixes of length at most *l* of the original test sequences. The practicality of the methods is also improved in comparison to the unbounded case: the size of the test suites may be considerably reduced while the complexity of the test generation algorithms remains basically unchanged.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

State-based languages, such as Statecharts [11] and SDL [19] are widely used for modelling systems which have an internal state, such as communications protocols and embedded control systems. As testing is a vital part of system development, this has led to much interest in testing from *finite state machines (FSMs)* [5,9,22,28,29,31,30,32]. Given a FSM specification, for which we have its transition diagram, and an implementation, which is a “black box” for which we can only observe its input/output behaviour, we want to test whether the implementation under test (IUT) conforms to the specification. This is called *conformance testing* or *fault detection* and a set of sequences that solves this problem is called a *test suite*.

Many test selection methods have been developed for the case in which the specification is a *deterministic FSM (DFSM)*. Most methods are based on the assumption that the implementation cannot have more states than the specification; among these, the best known are Transition Tour [31], (Multiple) Unique Input Output (UIO) [31,30], Characterizing Sequence [28] and Distinguishing Sequence [31,32] (the enumeration is in increasing order of their fault detection capability [29]). Although the condition is quite restrictive, these methods have been successfully used in testing of network protocols [31, 30,29]. A less restrictive condition is required by the *W*-method [5,31] and its variant, the “partial *W*” (*Wp*) method [9], which can be used for implementations that may have more states than the specification. The *W*- and *Wp*-methods generate test suites which guarantee the correctness of the IUT provided that the number of states of the IUT remain below a known upper bound. More recently, the *Wp*-method has been extended to generate timed test cases from Timed Finite State Machines [7].

When the specification is a DFSM, equivalence is the usual notion of correctness considered and most existing methods of testing from DFSMs (including the *W*- and *Wp*-methods) check that the IUT behaves as specified for *all* input sequences.

* Tel.: +40 21 4108964; fax: +40 248 216448.

E-mail address: florentin.ipate@ifsoft.ro.

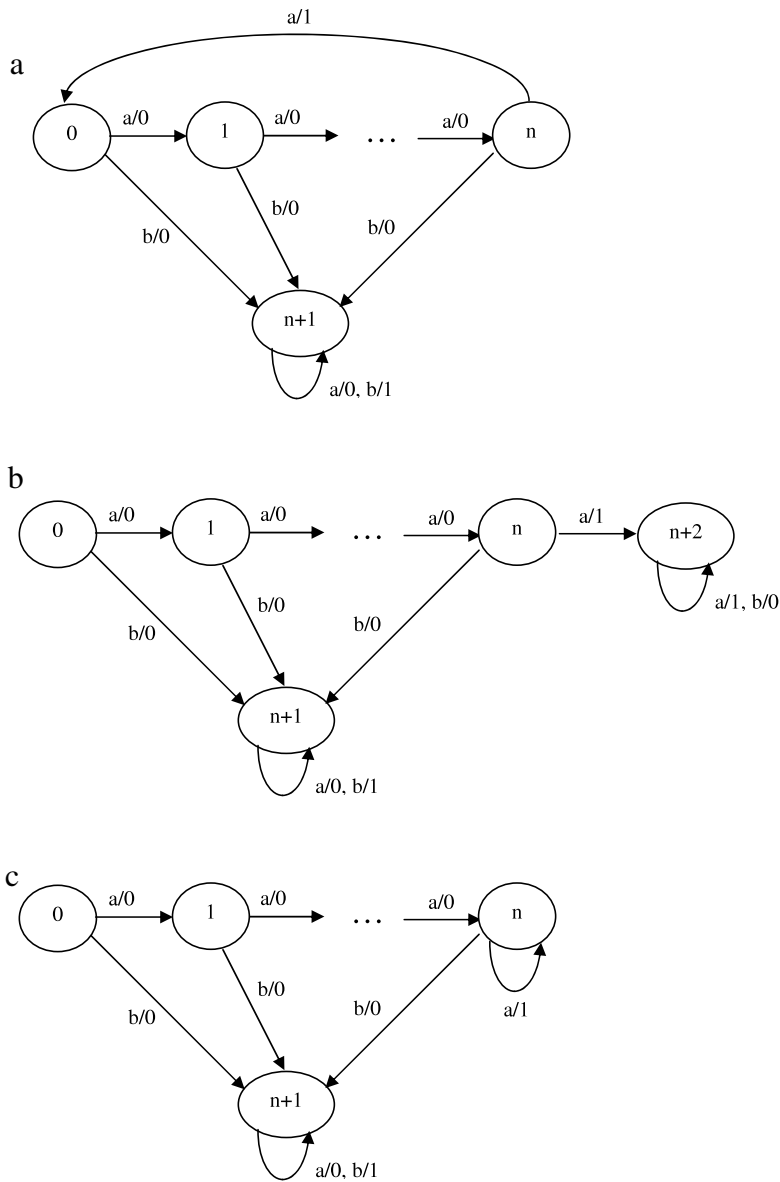


Fig. 1. Transition diagrams of M (a), M'_1 (b), and M'_2 (c).

On the other hand, in many applications, only input sequences of limited length are used. In such cases, the test suite only needs to establish that the IUT produces the specified results in response to input sequences whose length does not exceed an upper bound l . Naturally, the test suite produced in this case will only contain input sequences of length at most l .

Consider, for example, a simple device with two inputs, a and b , and two outputs, 0 and 1, which processes sequences of at most $n + 1$ characters. The device produces a 0 in response to any input except (i) when the input is the $(n + 1)$ th consecutive a in the sequence or (ii) when the input is a b , but not the first b in the sequence. We assume that the device is always reset after receiving $n + 1$ symbols, so the response to subsequent inputs can be ignored. This example will be used for illustration throughout the paper. The behaviour of this device can be described by a DFSM M as represented in Fig. 1(a) and an upper bound $l = n + 1$. On the other hand, M'_1 and M'_2 as represented in Fig. 1(b) and (c), respectively, produce the same outputs as M in response to any input sequence of length at most $l = n + 1$, so they are also valid models of this behaviour. However, none of them is equivalent to M (and, furthermore, they are not equivalent to one another), so a test suite that checks machine equivalence will be too strong in this case.

This paper extends the W - and Wp -methods to the case in which only bounded sequences are considered. The methods for bounded sequences are stronger than the originals since test suites for the unbounded case can be obtained as a particular case (in which the upper bound l is sufficiently large) from the new formulae. Furthermore, the generalization is not straightforward as it is not sufficient to extract the sequences of length at most l from the test suites produced in the

unbounded case, or even all prefixes of length at most l of the original test sequences. The practicality of the methods is also improved in comparison to the unbounded case: the size of the test suites may be considerably reduced while the complexity of the test generation algorithms remains basically unchanged. As a prerequisite for the new test generation methods, the paper also defines the concept of l -minimal DFSM.

Bounded conformance testing can be regarded as a middle ground between traditional FSM conformance testing, where the detection of all possible faults is sought, and ATPG (automatic test pattern generation) techniques [23], which are used to detect the erroneous behaviour caused by a particular fault. The use of bounded sequences for detecting faults has also been adopted by the formal verification community. Bounded Model Checking (BMC) based on SAT methods, [3,27], is rapidly gaining popularity as a complementary technique to BDD-based symbolic model checking. The general consensus in the community is that BMC works particularly well on large designs where bugs need to be searched at shallow to medium depths. Bounded conformance testing can be used to complement such verification techniques. Even where the design has been formally verified, the implementation still needs to be validated by functional testing.

The paper is structured as follows. Section 2 introduces finite state machine related concepts and results to be used later in the paper, while Section 3 presents the (original) W - and Wp -methods (for unbounded sequences). The following five sections address bounded sequence test generation: Section 4 formalizes the problem and shows that it cannot be solved using the original W - and Wp -methods; Section 5 introduces the concept of l -minimal DFSM, while the next two sections provide the new methods, for bounded sequences. The test generation process is illustrated with an example in Section 8. The formal proofs that validate the new test suites are given in the next two sections, by first constructing the l -product DFSM of the specification and IUT (Section 9) and then applying a state-counting strategy on this product machine (Section 10). Implementation and complexity issues are discussed in Section 11. Conclusions are drawn and future work is outlined in the final section.

2. Finite state machines

In this section we briefly introduce basic finite state machine concepts and results to be used in our presentation.

First, the notation used is introduced. For a finite set A , we use A^* to denote the set of finite sequences with members in A ; ϵ denotes the empty sequence. For $a, b \in A^*$, ab denotes the concatenation of sequences a and b . a^n is defined by $a^0 = \epsilon$ and $a^n = a^{n-1}a$ for $n \geq 1$. For $U, V \subseteq A^*$, $UV = \{ab \mid a \in U, b \in V\}$; U^n is defined by $U^0 = \{\epsilon\}$ and $U^n = U^{n-1}U$ for $n \geq 1$. Also, $U[n] = \cup_{0 \leq k \leq n} U^k$. For a sequence $a \in A^*$, $\|a\|$ denotes the number of elements of a ; in particular $\|\epsilon\| = 0$.

A *deterministic finite state machine (DFSM)* M is a tuple $(\Sigma, \Gamma, Q, h, q_0)$, where

- Σ is the finite *input alphabet*,
- Γ is the finite *output alphabet*,
- Q is the finite *set of states*,
- $h : Q \times \Sigma \rightarrow Q \times \Gamma$ is the (partial) *next-state and output function*;
- $q_0 \in Q$ is the *initial state*.

A DFSM is usually described by a state-transition diagram, see for example Fig. 1.

M is said to be *completely specified* if h is a total function. Otherwise M is said to be *partially specified*. In this case, the undefined value of h is denoted by \perp .

The (partial) function $h : Q \times \Sigma \rightarrow Q \times \Gamma$ breaks up into two (partial) functions: $h_1 : Q \times \Sigma \rightarrow Q$ and $h_2 : Q \times \Sigma \rightarrow \Gamma$ having a common domain. h_1 is called the *next-state function* and h_2 the *output function*. The next-state function h_1 can be extended to a (partial) function $h_1^* : Q \times \Sigma^* \rightarrow Q$ defined by: $h_1^*(q, \epsilon) = q$, $q \in Q$; $h_1^*(q, s\sigma) = h_1(h_1^*(q, s), \sigma)$, $q \in Q$, $s \in \Sigma^*$, $\sigma \in \Sigma$. The output function h_2 can be extended to a (partial) function $h_2^* : Q \times \Sigma^* \rightarrow \Gamma^*$ defined by: $h_2^*(q, \epsilon) = \epsilon$, $q \in Q$; $h_2^*(q, s\sigma) = h_2^*(q, s)h_2(h_1^*(q, s), \sigma)$, $q \in Q$, $s \in \Sigma^*$, $\sigma \in \Sigma$.

Given $q \in Q$, the (partial) function computed by M in q , $f_M^q : \Sigma^* \rightarrow \Gamma^*$, is defined by: $f_M^q(s) = h_2^*(q, s)$, $s \in \Sigma^*$. The function computed by M in q_0 is simply called the *function computed by M* and is denoted by f_M .

Given $p, q \in Q$ and $s \in \Sigma^*$, s is said to *reach* q from p if $h_1^*(p, s) = q$. $q \in Q$ is said to be *reachable* if there exists $s \in \Sigma^*$ that reaches q from the initial state q_0 . M is said to be *reachable* if all the states of M are reachable.

Given $Y \subseteq \Sigma^*$, two states $q_1, q_2 \in Q$ are said to be *Y-equivalent* if for all $s \in Y$, $h_2(q_1, s) = h_2(q_2, s)$ (this includes the case in which $h_2(q_1, s) = h_2(q_2, s) = \perp$). Otherwise q_1 and q_2 are said to be *Y-distinguishable*. If $Y = \Sigma^*$, q_1 and q_2 are simply said to be *equivalent* or *distinguishable*. Two DFSMs are said to be (Y-)equivalent or (Y-)distinguishable if their initial states are (Y-)equivalent or (Y-)distinguishable. M is said to be *reduced* if every two distinct states of M are distinguishable.

M is called *minimal* if any DFSM that computes f_M has at least the same number of states as M . Then M is minimal if and only if M is reachable and reduced. Furthermore, given a DFSM M , there is a unique (up to a renaming of its states) minimal DFSM that computes f_M . More formally, two DFSMs $M = (\Sigma, \Gamma, Q, h, q_0)$ and $M' = (\Sigma, \Gamma, Q', h', q'_0)$ are said to be *isomorphic* if there is a bijective function $g : Q \rightarrow Q'$ with $g(q_0) = q'_0$ such that $g(h_1(q, \sigma)) = h'_1(g(q), \sigma)$ and $h_2(q, \sigma) = h'_2(g(q), \sigma)$, $q \in Q$, $\sigma \in \Sigma$ (in particular $h(q, \sigma) = \perp$ if and only if $h'(g(q), \sigma) = \perp$). Then, for two minimal DFSMs M and M' , $f_M = f_{M'}$ if and only if M and M' are isomorphic. For proofs of these well-known results and for techniques for constructing the minimal DFSM equivalent to a given DFSM see for example [6,10] or [14].

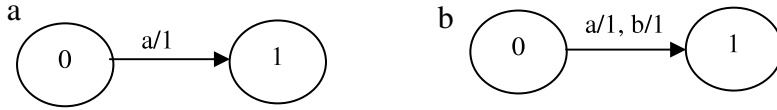


Fig. 2. Transition diagrams of M_0 (a) and M'_0 (b).

3. The W - and Wp -methods

This section briefly reviews the W - and Wp -methods for generating test suites from a DFSM specification. The DFSM specification M considered is assumed to be completely specified (there is a transition on every input from every state) and minimal (all states are reachable from the initial state and there are no equivalent states).

When testing from a DFSM (and, in general, from a formal specification) it is usual to assume that the IUT behaves like some unknown element from a *fault domain*. In the case of the W - and Wp -methods, the fault domain consists of all completely specified DFSMs with the same input alphabet Σ and output alphabet Γ as M , whose number of states m' does not exceed the number of states m of M by more than k ($m' - m \leq k$), where $k \geq 0$ is a predetermined integer. Furthermore, it is assumed that the IUT has a reliable reset. A DFSM has a reset operation if there is some input r that takes every state to the initial state. A reliable reset is a reset that is known to have been implemented correctly. The simplest way to obtain a reliable reset is through the system being switched off and then on again. The reset will not be included in the input alphabet. A *test suite* will be a finite set Y of input sequences that, for every M' in the fault model that is not equivalent to M , will produce at least one erroneous output. That is, M and M' are equivalent whenever M and M' are Y -equivalent.

3.1. The W -method

The W -method involves the selection of two sets of input sequences, S and W as follows:

- The set $S \subseteq \Sigma^*$, called a *state cover* of M , will contain sequences that reach all states of M ; in particular, S will contain the empty sequence ϵ – this will be used to reach the initial state. For M as represented in Fig. 1(a), ϵ will reach 0, a will reach 1, \dots , a^n will reach n , while b will reach $n + 1$. Thus $S = \{\epsilon, a, \dots, a^n, b\}$ is a state cover of M .
- The set $W \subseteq \Sigma^*$, called a *characterization set* of M , will distinguish between any two distinct states of M . That is, given any two distinct states q and q' of M , W will contain at least one input sequence that produces different output sequences when applied to q and q' , respectively. For M as represented in Fig. 1(a), consider the application of the input sequence $a^{n-1}b$ to each state of M . The output sequence produced in states 0 and 1 will be 0^n ; for $2 \leq j \leq n$, the output produced in state j will be $0^{n-j}10^{j-1}$; the output sequence produced in state $n + 1$ will be $0^{n-1}1$. Thus, $a^{n-1}b$ will distinguish between any pair of distinct states of M except (0, 1). On the other hand, 0 and 1 will be distinguished by a^n . Indeed, the application of the input sequence a^n in state 0 will produce the output sequence 0^n , while the application of the same input sequence in state 1 will produce the output sequence $0^{n-1}1$. Thus $W = \{a^n, a^{n-1}b\}$ is a characterization set of M .

Once S and W have been constructed, a test suite is generated using the formula

$$U_k = S\Sigma[k + 1]W,$$

where, for $n \geq 0$, $\Sigma[n] = \bigcup_{i=0}^n \Sigma^i$ is the set of all input sequences of length less than or equal to n .

The idea is that the set $P = S \cup S\Sigma$ (usually called a *transition cover* of M) ensures that all the states and all the transitions of M are also present in M' and the set $\Sigma[k]W$ ensures that the destination state of each transition of M' is also correct. Notice that the latter contains W and also all sets $\Sigma^i W$, $1 \leq i \leq k$. This ensures that M' does not contain extra states; if there were up to k extra states, each of them would be reached by some input sequence of up to length k from the existing states.

Interestingly, the test suite given above ensures that the IUT M' is equivalent to the specification M when these are completely specified DFSMs, but this may not necessarily be true if M and M' are partially specified. In such DFSMs, there will be states from which not every input will have a corresponding transition. In this case, the output produced by an input sequence that would fire any such missing transition will be considered to be “*undefined*” – a special value, different from the normal outputs. Consider for example the specification M_0 as represented in Fig. 2(a) and the DFSM model of the IUT M'_0 as represented in Fig. 2(b). M_0 and M'_0 are non-equivalent partially specified DFSMs. $S = \{\epsilon, a\}$ is a state cover of M_0 and $W = \{a\}$ is a characterization set of M_0 . Since M_0 and M'_0 have the same number of states (2), the W -method gives $U_0 = S\Sigma[1]W = \{a, aa, ba, aaa, aba\}$. It can be observed that b is the only input sequence for which the output produced by M_0 (*undefined*) is different from the output produced by M'_0 (1), but this is not contained in U_0 .

Intuitively, this happens because, when M and M' are partially specified, they may produce identical (*undefined*) outputs on some input sequence s but different outputs (one defined and one *undefined*) on some prefix t of s . In our example, the output produced by the sequence ba , which is contained in U_0 , is *undefined* for both M and M' , whereas b produces the *undefined* output on M , but 1 on M' . Consequently, a solution would be to take the set of all prefixes of U_k instead of just U_k . In [1] it is shown that only a subset of the set of all prefixes is required; this is:

$$U'_k = S\Sigma[k + 1]W_\epsilon \setminus \{\epsilon\},$$

where $W_\epsilon = W \cup \{\epsilon\}$. In our example $U'_0 = \{a, b, aa, ab, ba, aaa, aba\}$, so $b \in U'_0$.

Note that, in the above discussion, as often in practice, the missing transitions of a DFSM are assumed to represent erroneous behaviour. That is, the “refused” inputs are assumed to produce a designated error output, which is not in the output alphabet of M . In this case, a DFSM may be transformed into one that is completely specified by representing the erroneous behaviour as self-looping transitions or transitions to an extra (error) state [22] and the original W -method can be applied to the newly obtained machine. In the above discussion, however, we considered the case in which test suites are directly derived from the original, partially defined specification. An alternative semantics is to consider that the missing transitions are not relevant for the specified system, that is the refused inputs may produce any of the symbols in the output alphabet of M . In this case, the specification M and the IUT need not be equivalent; instead, a fault-free IUT M' would be a completely specified DFSM M' that contains the behaviour specified by M and also additional behaviour, that replaces the missing transitions. Test generation from partially specified DFMS in this context is addressed in [26].

3.2. The Wp -method

The *partial W -method* (or *Wp -method*) [9] is an improvement of the W -method that may reduce the size of the test suite at the expense of a slightly more complex generation algorithm. Instead of using the whole set W to check the destination state q of each transition, only a subset W_q of this set is used. This subset, which depends on the reached state q , is called an *identification set of q* and distinguishes q from any other state of M . The set $\mathcal{W} = \{W_q \mid q \in Q\}$ of all identification sets used (one for each state) is called an *identification set of M* . Naturally, the union of all identification sets $W_q \in \mathcal{W}$ will make up a characterization set of M .

Now, suppose we have a state cover S and a characterization W set of the specification M , as above. Let $R = S\Sigma \setminus S$. Additionally, we construct an identification set \mathcal{W} of M such that each $W_q \in \mathcal{W}$ is contained in W , $W_q \subseteq W$. Then, in the case of completely specified DFSMs, the test suite produced by the Wp -method is given by the formula

$$V_k = S\Sigma[k]W \cup R\Sigma[k] \otimes \mathcal{W},$$

where, for a set A of input sequences, $A \otimes \mathcal{W}$ consists of each s in A concatenated with the corresponding W_q such that q is reached by s , i.e. $A \otimes \mathcal{W} = \bigcup \{sW_q, s \in A, W_q \in \mathcal{W}, s \text{ reaches } q\}$.

Intuitively, the first component $V_k^1 = S\Sigma[k]W$ checks that all the states defined by the specification are present in the implementation. At the same time, the transitions leading from the initial state to these states are checked for correct output and state transfer. The second component $V_k^2 = R\Sigma[k] \otimes \mathcal{W}$ checks the implementation for all the transitions that are not checked by V_k^1 . Since all the sets W_q are contained in W , the resulting test suite V_k is contained in the set U_k produced by the W -method. As some of the identification sets may be strictly contained in W , V_k may have less elements than U_k .

For M in our example, the characterization set chosen was $W = \{a^n, a^{n-1}b\}$. As pointed out earlier, $a^{n-1}b$ distinguishes between any pair of distinct states of M except $(0, 1)$. Thus, for $2 \leq j \leq n+1$, $W_j = \{a^{n-1}b\}$ is an identification set of j . Consider now the application of a^n to each state of M . The output sequence produced in states 0 and $n+1$ will be 0^n , while, for $1 \leq j \leq n$, the output produced in state j will be $0^{n-j}10^{j-1}$. Thus a^n distinguishes 1 from any other state and so $W_1 = \{a^n\}$ is an identification set of 1. On the other hand, both a^n and $a^{n-1}b$ are required to distinguish 0 from the other states of M , thus $W_0 = W = \{a^n, a^{n-1}b\}$. By definition, $\mathcal{W} = \{W_0, W_1, \dots, W_n\}$ is an identification set of M . The state cover chosen was $S = \{\epsilon, a, \dots, a^n, b\}$ and so $R = S\Sigma \setminus S = \{a^{n+1}\} \cup \{b, ab, \dots, a^n b, ba, bb\}$. Then $V_0 = SW \cup R \otimes \mathcal{W} = SW \cup \{a^{n+1}\}W_0 \cup \{b, ab, \dots, a^n b, ba, bb\}W_{n+1}$.

Analogously to the W -method, the Wp -method can be extended to cope with partially specified DFSMs [2]. In this case, the revised formula for the test suite is

$$V'_k = S\Sigma[k]W_\epsilon \cup R\Sigma[k] \otimes \mathcal{W}_\epsilon \setminus \{\epsilon\},$$

where, for a set A of input sequences, $A \otimes \mathcal{W}_\epsilon = (A \otimes \mathcal{W}) \cup A$.

For M_0 and M'_0 as represented in Fig. 2, $R = \{b, aa, ab\}$, $W_0 = W_1 = W = \{a\}$, $\mathcal{W} = \{\{a\}, \{a\}\}$, $R \otimes \mathcal{W} = \emptyset$, $R \otimes \mathcal{W}_\epsilon = \{b, aa, ab\}$, so $V_0 = \{a, aa\}$ and $V'_0 = \{a, b, aa, ab\}$. Since $b \in V'_0 \setminus V_0$, the fault in the implementation will be detected by V'_0 but not by V_0 .

4. Bounded sequence test generation – preliminaries

In this section we show how the W - and Wp -methods can be extended to the case of bounded sequences. In this case, the test suite will contain only sequences of length less than or equal to an upper bound $l \geq 1$ and will establish if the IUT behaves as specified for every sequence in $\Sigma[l]$.

The extension to the bounded case is not straightforward, since it is not sufficient to extract the sequences of length at most l from the test suites that establish the equivalence of the two DFSMs, or even all prefixes of length at most l of these test sequences. Consider again M as represented in Fig. 1(a) and $l = n+1$, $n \geq 2$. As shown earlier, $S = \{\epsilon, a, \dots, a^n, b\}$ is a state cover of M and $W = \{a^n, a^{n-1}b\}$ is a characterization set of M . Thus $U_0 = S\Sigma[1]W = \{\epsilon, a, \dots, a^n, b\}\{\epsilon, a, b\}\{a^n, a^{n-1}b\}$. Consider M' as represented in Fig. 3, the DFSM model of a faulty implementation. The only sequences of length less than or equal to $n+1$ that distinguish between M and M' are the sequences that contain at least three b s. The only sequence in U_0 that contains three b s is $bba^{n-1}b$. However, any prefix of length less than or equal to $n+1$ of this sequence will contain at most two b s. Thus, by extracting all prefixes of length less than or equal to $n+1$ of the sequences in U_0 and applying the obtained set to the implementation, no fault will be detected. Furthermore, consider the extended test suite used in the case

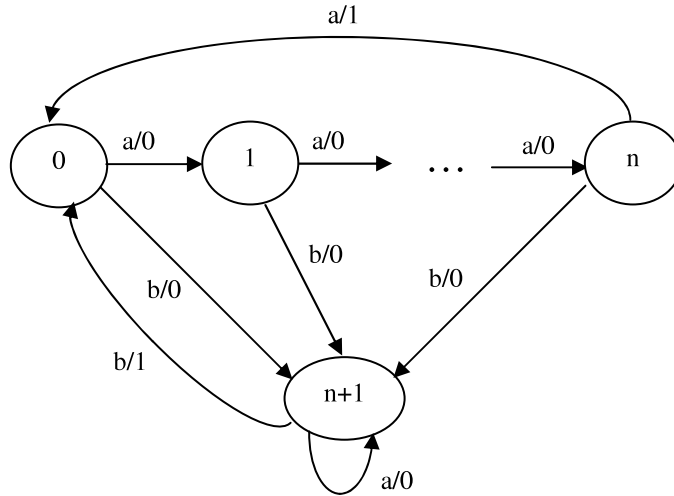


Fig. 3. Transition diagram of M' .

of partially specified DFSMs, $U'_0 = S\Sigma[1]W_\epsilon \setminus \{\epsilon\} = \{\epsilon, a, \dots, a^n, b\}\{\epsilon, a, b\}\{\epsilon, a^n, a^{n-1}b\} \setminus \{\epsilon\}$. It can be observed that any prefix of length less than or equal to $n + 1$ of some sequence in U'_0 will contain at most two bs , so still no fault will be found.

On the other hand, the underlying ideas behind the W - and Wp -methods can also be applied in the case of bounded sequences, provided that the sets S , W , and \mathcal{W} used in the construction of the test suites will contain sequences of *minimum length* that reach or distinguish the states of M . The new test suites will be given in Sections 6 and 7, respectively.

First, recall that in the unbounded case the DFSM specification used for test generation was required to be minimal. As, naturally, this requirement will be maintained, the concept of minimal DFSM needs to be redefined for the bounded case. The formal proofs that validate the construction of the new test suites are given in Sections 9 and 10.

5. l -minimal DFSMs

The concept of l -minimal DFSM introduced here is a natural extension of the minimal deterministic finite cover automaton (DFCA) of a finite language defined in [4]. Given $l \geq 1$, a DFSM is said to be l -minimal if any DFSM that behaves identically to M for every sequence of length less than or equal to l has at least as many states as M .

Definition 5.1. Given $l \geq 1$, a DFSM M is said to be l -minimal, if any DFSM $\Sigma[l]$ -equivalent to M has at least the same number of states as M .

Recall that, when sequences of unlimited length are considered, a minimal DFSM is a DFSM in which all states are reachable and pairwise distinguishable. Analogously, the states of an l -minimal DFSM need be reachable and distinguishable by sequences of length at most l .

More precisely, given a state q of M , $level_M(q)$ is defined to be the length of the shortest input sequence(s) that reach q . Then, in order for M to be l -minimal, every two distinct states q_1 and q_2 must be distinguished by some input sequence of length at most $l - \max\{level_M(q_1), level_M(q_2)\}$ (otherwise a DFSM $\Sigma[l]$ -equivalent to M with less states than M can be constructed). If this is the case, q_1 and q_2 will be said to be l -dissimilar; otherwise q_1 and q_2 will be said to be l -similar. Then a DFSM M is l -minimal if and only if all states of M are reachable and pairwise l -dissimilar. These concepts and results are formally expressed in what follows.

Definition 5.2. Let $M = (\Sigma, \Gamma, Q, h, q_0)$ be a reachable DFSM. For each state $q \in Q$ we define $level_M(q)$ as the length of the shortest path(s) from q_0 to q , i.e.

$$level_M(q) = \min\{\|s\| \mid s \in \Sigma^*, h_1^*(q_0, s) = q\},$$

where $\|s\|$ denotes the number of elements of the sequence s .

We use again for illustration M , M'_1 and M'_2 as represented in Fig. 1(a), (b) and (c), respectively, $n \geq 1$, and $l = n + 1$. For example, $level_M(i) = i$ for every state i of M , $0 \leq i \leq n$ and $level_M(n + 1) = 1$.

Definition 5.3. Let $M = (\Sigma, \Gamma, Q, h, q_0)$ be a reachable DFSM and $l \geq 1$. Then \sim_M^l is a relation on Q defined by: $p \sim_M^l q$ if p and q are $\Sigma[n]$ -equivalent, where $n = \min\{l - level_M(p), l - level_M(q), 0\}$. We say that p is l -similar to q ; otherwise p and q are said to be l -dissimilar.

It can be observed that for every states i and j of M , $0 \leq i < j \leq n$, i and j are distinguished by a^{n+1-j} . Thus, since $\min\{n + 1 - level_M(i), n + 1 - level_M(j)\} = n + 1 - j$, i and j are $(n + 1)$ -dissimilar. Furthermore, state $n + 1$ is $(n + 1)$ -dissimilar to any other state i , $0 \leq i \leq n$, since it can be distinguished from i by b and $\min\{n + 1 - level_M(n + 1), n + 1 - level_M(i)\} =$

$\min\{n, n + 1 - i\} \geq 1$. On the other hand, not every pair of states of M'_1 is $(n + 1)$ -dissimilar; as $level_{M'_1}(n + 2) = n + 1$, state $n + 2$ is $(n + 1)$ -similar to any other state of M'_1 .

Note that, unlike equivalence, l -similarity is not a transitive relation. State $n + 2$ is $(n + 1)$ -similar to any other state of M'_1 , but all the remaining pairs of states are $(n + 1)$ -dissimilar.

Definition 5.4. A reachable DFSM M is called l -reduced if every two distinct states of M are l -dissimilar.

Theorem 5.1. A DFSM M is l -minimal if and only if it is reachable and l -reduced.

Proof. “ \Rightarrow ” If M was not reachable then the unreachable states could be removed without affecting the function computed by M , so M must be reachable. Assume M is not l -reduced and let q_1 and q_2 be two l -dissimilar states of M . Assume $level_M(q_1) \leq level_M(q_2)$. Then we can construct M' , a DFSM $\Sigma[l]$ -equivalent to M by removing q_2 and replacing all transitions that led to q_2 in the original DFSM with transitions to q_1 (further details are omitted since a similar construction is given in [4]). As M' has one state less than M , M is not l -minimal, which contradicts our original hypothesis.

“ \Leftarrow ” Assume M is reachable and l -reduced but not l -minimal. For each state q of M , let x_q denote an input sequence of minimum length that reaches q (that is, $h_1^*(q_0, x_q) = q$ and $\|x_q\| = level_M(q)$). As M is not minimal, there must be a DFSM $M' = (\Sigma, \Gamma, Q', h', q'_0)$ with less states than M that is $\Sigma[l]$ -equivalent to M . As M' has less states than M , there exist two states p and q of M for which $h_1^*(q'_0, x_p) = h_1^*(q'_0, x_q)$. As M and M' are $\Sigma[l]$ -equivalent, it follows that p and q are l -similar, which contradicts our original hypothesis. \square

In our example, M and M'_2 are $(n + 1)$ -minimal since their states are reachable and pairwise $(n + 1)$ -dissimilar. On the other hand, M'_1 is not $(n + 1)$ -minimal since it is not $(n + 1)$ -reduced. It can be observed that the $(n + 1)$ -minimal DFSM M'_2 will be obtained by removing state $n + 2$ from M'_1 and replacing it with n as destination state in the transitions of M'_1 .

A direct consequence of Theorem 5.1 is that every l -minimal DFSM is also a minimal DFSM.

Corollary 5.1. If a DFSM is l -minimal for some $l \geq 1$ then M is a minimal DFSM.

Proof. Follows from Theorem 5.1 since any two states that are l -dissimilar are also distinguishable. \square

On the other hand, given an upper bound l , a DFMS may be minimal but not l -minimal. This is illustrated by the above example: M'_1 is minimal, but not $(n + 1)$ -minimal.

Unlike in the unbounded case, there may be several distinct l -minimal DFSMs that are $\Sigma[l]$ -equivalent. In our example, M and M'_2 are $\Sigma[n + 1]$ -equivalent and are both $(n + 1)$ -minimal but they are not isomorphic (it is not possible to obtain one from the other by renaming the states).

Since the concept of l -minimal DFSM introduced here is an extension of the minimal deterministic finite cover automaton (DFCA) of a finite language defined in [4], the algorithms for constructing minimal DFCA's provided in [4,24,21] and [15] can be naturally extended to DFSMs. The algorithms for constructing a minimal DFCA given in [4,24,21] have time complexity $O(m^4)$, $O(m^2)$ and $O(m \log m)$, respectively, where m is the number of states of the original automaton. The algorithm given in [4] is extended in [15] to determine all minimal DFCA's of a finite language. A detailed presentation of these algorithms or of the way they can be adapted to construct l -minimal DFSMs is, however, beyond the scope of this paper.

6. The W -method for bounded sequences

Suppose the specification used for test generation is an l -minimal DFSM M (the general case in which M may be partially specified is considered). The fault domain consists of all (possibly partially specified) DFSMs with the same input alphabet Σ and output alphabet Γ as M whose number of states m' does not exceed the number of states m of M by more than k .

The construction of the test suite will involve the selection of a proper state cover and of a strong characterization set of M , defined in what follows.

A proper state cover S is a set of input sequences which, for every state q of M , contains a sequence of minimum length ($level_M(q)$) that reaches q .

Definition 6.1. $S \subseteq \Sigma^*$ is called a proper state cover of M if for every state q of M there exists $s \in S$ such that $h_1^*(q_0, s) = q$ and $\|s\| = level_M(q)$.

Note that, since the states of M are pairwise l -dissimilar, $level_M(q) \leq l - 1$ for every state q of M , so a minimal proper state cover will contain sequences of length less than or equal to $l - 1$. For M as represented in Fig. 1(a), state i , $0 \leq i \leq n$, is reached by a^i , but by no shorter sequences. Similarly, b is the shortest sequence to reach state $n + 1$. Thus $S = \{\epsilon, a, \dots, a^n\}$ is a proper state cover of M .

A strong characterization set is a set of input sequences which, for every two distinct states q_1 and q_2 of M , contains a sequence of minimum length that distinguishes q_1 and q_2 .

Definition 6.2. $W \subseteq \Sigma^*$ is called a strong characterization set of M if for every two states q_1 and q_2 of M and every $j > 0$, if q_1 and q_2 are $\Sigma[j]$ -distinguishable then q_1 and q_2 are $(W \cap \Sigma[j])$ -distinguishable.

Naturally, in the above definition, it is sufficient for q_1 and q_2 to be $(W \cap \Sigma[j])$ -distinguishable when j is the length of the shortest sequences that distinguish between q_1 and q_2 .

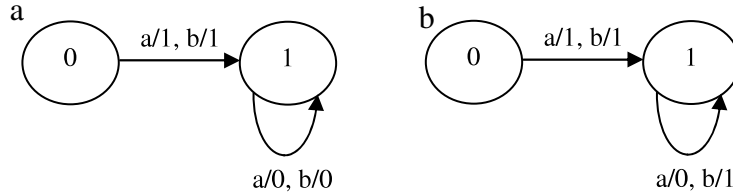


Fig. 4. Transition diagrams of M^1 (a) and M^2 (b).

For M in our example, states i and j , $0 \leq i < j \leq n$, are distinguished by a^{n+1-j} , but by no sequence of length less than $n+1-j$. On the other hand, b distinguishes $n+1$ from any other state of M . Thus $W = \{a, \dots, a^n\}$ is a strong characterization set of M .

Once S and W have been selected, the test suite for the bounded case is obtained using the formula:

$$Y_k = S\Sigma[k+1]W_\epsilon \cap \Sigma[l] \setminus \{\epsilon\}.$$

This result will be formally proven later (Theorem 10.3). For M in our example and $l = n+1$, $Y_0 = S\Sigma[1]W_\epsilon \cap \Sigma[n+1] \setminus \{\epsilon\} = \{a, \dots, a^{n+1}\} \cup \{a^i b a^j \mid 0 \leq i \leq n, 0 \leq j \leq n-i\} \cup \{a^i b b \mid 0 \leq i \leq n-1\} \cup \{b b a^i \mid 1 \leq i \leq n-1\} \cup \{b a b, b b b\}$. Consider again M' as represented in Fig. 3, $n \geq 2$. As $b b b \in Y_0$, Y_0 will distinguish between M and M' .

Note that Y_k is also a valid test suite for partially specified DFSMs. Consider M_0 and M'_0 as represented in Fig. 2 and $l = 2$. $S = \{\epsilon, a\}$ is a proper state cover of M_0 and $W = \{a\}$ is strong characterization set of M_0 . Then $Y_0 = S\Sigma[1]W_\epsilon \cap \Sigma[2] \setminus \{\epsilon\} = \{a, b, a a, a b, b a\}$. Since $b \in Y_0$, Y_0 will detect the fault in M' .

On the other hand, W_ϵ , rather than only W , is needed in the definition of Y_k even if the two DFSMs are completely specified. Consider M^1 and M^2 as represented in Fig. 4(a) and (b), respectively, and $l = 2$. Then $a b$ and $b b$ are the only sequences of length less than or equal to 2 that distinguish between M^1 and M^2 . $S = \{\epsilon, a\}$ is a proper state cover of M^1 and $W = \{a\}$ is a strong characterization set and so $S\Sigma[1]W \cap \Sigma[2] = \{a, a a, b a\}$, which contains neither of the two distinguishing sequences. Thus, if W was used instead of W_ϵ in the formula of Y_k , no fault would be detected.

The test suite is generated using a state-counting approach (see Section 10). State-counting, which was originally used in [25] for conformance testing of a deterministic implementation against a non-deterministic FSM (see also [12]), involves the construction of the product FSM of the specification and of the (unknown) implementation. When only sequences of length at most l are considered, an l -bounded product FSM is constructed instead (see Section 9). A test suite is then generated using a breadth-first search through input/output sequences in which the termination criterion is based on the observation that if a pair of states (q of M , q' of M'), from which a failure may be exhibited, is reachable then it is reachable by some minimal input/output sequence. Such a minimal sequence will not have visited identical or equivalent (with respect to sequences of length at most l) pairs of states and cannot contain pairs of states that have already been reached by the sequences in S . The strong characterization set is used to determine when a sequence has met identical or equivalent pairs, so it can be pruned. The traversal of the reachable space of the product machine is also used in formal verification of finite state machines [8], but the transition graph of the implementation is known there and can also be used in partitioning the state space [20].

7. The Wp -method for bounded sequences

Similarly, the Wp -method will also select the shortest possible sequences to distinguish between the states of M . A strong identification set of state q is defined as a set of input sequences which, for every other state q' of M , contains a sequence of minimum length that distinguishes q and q' .

Definition 7.1. Given $q \in Q$, $W_q \subseteq \Sigma^*$ is called a strong identification set of q if for every state q' of M and every $j > 0$, if q and q' are $\Sigma[j]$ -distinguishable then q and q' are $(W \cap \Sigma[j])$ -distinguishable. A set $\mathcal{W} = \{W_q \mid q \in Q\} \subseteq 2^{\Sigma^*}$ that contains a strong identification set W_q of q for each state q of M is called a strong identification set of M .

Suppose we have constructed

- a proper state cover S ,
- a strong characterization set W ,
- a strong identification set $\mathcal{W} = \{W_q \mid q \in Q\}$ of M such that $W_q \subseteq W$ for every $q \in Q$

and have determined $R = S\Sigma \setminus S$. Then, as shown later (Theorem 10.2), the test suite will be obtained using the formula:

$$Z_k = (S\Sigma[k]W_\epsilon \cup R\Sigma[k] \otimes \mathcal{W}_\epsilon) \cap \Sigma[l] \setminus \{\epsilon\}.$$

Consider again M and M' as represented in Fig. 1(a) and Fig. 3, respectively, and $l = n+1$, $n \geq 2$. $S = \{\epsilon, a, \dots, a^n, b\}$ is a proper state cover of M and so $R = S\Sigma \setminus S = \{a^{n+1}\} \cup \{b, a b, \dots, a^n b, b a, b b\}$. $W = \{a, \dots, a^n, b\}$ is a strong characterization set of M . $W_0 = \{a, \dots, a^n, b\}$ is a strong identification set of 0, for $1 \leq j \leq n-1$, $W_j = \{a, \dots, a^{n-j+1}, b\}$ is a strong identification set of j , $W_n = \{a\}$ is a strong identification set of n and $W_{n+1} = \{b\}$ is a strong identification set of $n+1$. Thus $\mathcal{W} = \{W_0, W_1, \dots, W_n\}$ is a strong identification set of M . Then $Z_0 = (S\Sigma \cup R \otimes \mathcal{W}_\epsilon) \cap \Sigma[n+1] \setminus \{\epsilon\} = (S\Sigma \cup \{a^{n+1}\})W_0 \cup$

$\{b, ab, \dots, a^n b, ba, bb\}W_{n+1} \cup R) \cap \Sigma[n+1] \setminus \{\epsilon\} = \{a, \dots, a^{n+1}\} \cup \{b, \dots, a^n b\} \cup \{bb, \dots, a^{n-1} bb\} \cup \{ba, bab, bbb\}$. As $bbb \in Z_0$, Z_0 will distinguish between M and M' .

As in the case of unbounded sequences, the test suite generated by the Wp -method may be strictly contained in the test suite generated by the W -method. In the above example, baa is contained in Y_0 but not in Z_0 .

Analogously to Y_k , Z_k is also a valid test suite in the case of partially specified DFSMs. For M_0 and M'_0 as represented in Fig. 2 and $l = 2$, $S = \{\epsilon, a\}$, $R = \{b, aa, ab\}$, $W_0 = W_1 = W = \{a\}$, $R \otimes \mathcal{W}_\epsilon = R = \{b, aa, ab\}$. Thus $Z_0 = SW_\epsilon \cup R \setminus \{\epsilon\} = \{a, b, aa, ab\}$. Since $b \in Z_0$, Z_0 will distinguish between M and M' .

Note that test suites for unbounded sequences can be immediately derived from the formulae for the bounded case. These are exactly the test suites produced by the original methods (given in Sections 3.1 and 3.2), with the mention that S , W and \mathcal{W} are required to be a *proper* state cover, a *strong* characterization set and a *strong* identification set, respectively. In the case of (possibly) partially specified DFSMs, U'_k and V'_k are obtained by simply removing the upper bound (i.e. considering l sufficiently large) from the formulae of Y_k and Z_k , respectively. For completely specified DFSMs, two additional observations are used: (i) every sequence in $U'_k \setminus U_k$ and $V'_k \setminus V_k$ is a prefix of some sequence in U_k and V_k , respectively; (ii) if two completely specified DFSMs are distinguished by some prefix of an input sequence, they are also distinguished by the entire sequence. Thus, if the specification and the IUT are known to be completely specified, the test suites U_k and V_k are sufficient. Formal proofs of these results will be given later (Corollaries 10.1 and 10.2).

8. Example

Before we proceed, we illustrate the test generation process for the DFSM specification N , as represented in Fig. 5(a), and $l = 5$. N is a minimal DFSM. However, states 0 and 4 are Σ -equivalent and since $level_N(4) = 4$, $0 \sim_N^4 4$. Similarly, since states 1 and 3 are $\Sigma[2]$ -equivalent and $level_N(3) = 3$, $1 \sim_N^3 3$. An l -minimal DFSM $\Sigma[l]$ -equivalent to N is N' , as represented in Fig. 5(b). The test generation methods for bounded sequences can now be applied to the l -minimal DFSM N' .

Since ϵ , a and ba are the shortest sequences that reach states 0, 1 and 2, respectively, $S = \{\epsilon, b, ba\}$ is a proper state cover of N' . States 0 and 1 are $\{a\}$ -distinguishable and states 1 and 2 are also $\{a\}$ -distinguishable. On the other hand, states 0 and 2 are $\{aa\}$ -distinguishable but Σ -equivalent. Thus, $W = \{a, aa\}$ is a strong characterization set of N' . For $k = 0$, the test set produced by the application of the W -method for bounded sequences is $Y_0 = S\Sigma[1]W_\epsilon \cap \Sigma[l] \setminus \{\epsilon\}$. Hence $Y_0 = \{a, b, aa, ba, bb, aaa, baa, bab, bba, baaa, baba, bbaa, baaaa, babaa\}$.

Consider now the application of the Wp -method for bounded sequences for $k = 0$. $R = S\Sigma \setminus S = \{a, bb, baa, bab\}$. From the above observations it follows that $W_0 = \{a, aa\}$, $W_1 = \{a\}$ and $W_2 = \{a, aa\}$ are strong identification sets of states 0, 1 and 2, respectively. Thus $Z_0 = (SW_\epsilon \cup R \otimes \mathcal{W}_\epsilon) \cap \Sigma[l] \setminus \{\epsilon\} = (SW_\epsilon \cup \{a, bab\}W_0 \cup \{bb, baa\}W_1 \cup R) \cap \Sigma[l] \setminus \{\epsilon\}$. Hence $Z_0 = \{a, b, aa, ba, baa, baaa\} \cup \{aa, aaa, baba, babaa\} \cup \{bba, baaa\} \cup \{a, bb, baa, bab\} = \{a, b, aa, ba, bb, aaa, baa, bab, bba, baaa, baba, babaa\}$.

9. l -bounded product machine

In order to establish the equivalence of two DFSMs, M and M' , one can build a cross-product of their states, such that states (q, q') of the cross-product DFSM correspond to pairs of states q, q' in the two DFSMs. A transition on input σ and output γ between states (q, q') and (q_1, q'_1) exists in the cross-product DFSM if and only if both transitions $h(q, \sigma) = (q_1, \gamma)$ and $h'(q', \sigma) = (q'_1, \gamma)$ exist in M and M' , respectively. The result of such a construction corresponds to the intersection (i.e. the set of all identical input sequence/output sequence pairs) of the functions computed by the two DFSMs. If f_M and $f_{M'}$ are different, the transitions from the corresponding states, q in M and q' in M' , will produce different outputs, i.e. $h_2(q, \sigma) \neq h'_2(q', \sigma)$. In such instances, the transition from (q, q') in the cross-product DFSMs will lead to an extra state, *Fail*. When at least one of M and M' may be partially specified, the transition from (q, q') will also lead to *Fail* when one of the transitions in the individual machines is defined and the other is not. On the other hand, when both individual transitions are not defined, the input sequence supplied to the machines need not be extended further; in this case, the transition in the cross-product DFSM will lead to another extra state, *Undef*.

When only the results produced by the two DFSMs in response to input sequences of length at most l are compared, an integer i , $1 \leq i \leq l$, can be added to the state space and incremented by each transition. No transition will be defined in the cross-product DFSM when $i = l$. The resulting construction will be called an l -bounded product DFSM of M and M' .

Definition 9.1. Given $l \geq 1$, the l -bounded product DFSM formed from $M = (\Sigma, \Gamma, Q, h, q_0)$ and $M' = (\Sigma, \Gamma, Q', h', q'_0)$ is the DFSM $P_l(M, M') = (\Sigma, \Gamma_p, Q_p, H, (q_0, q'_0, 0))$ in which $\Gamma_p = \Gamma \cup \{\text{fail}, \text{undef}\}$ with $\text{fail} \neq \text{undef}$ and $\text{fail}, \text{undef} \notin \Gamma$, $Q_p = Q \times Q' \times \{0, \dots, l\} \cup \{\text{Fail}, \text{Undef}\}$ with $\text{Fail} \neq \text{Undef}$ and $\text{Fail}, \text{Undef} \notin Q \times Q' \times \{0, \dots, l\}$ and H is defined by the following rules:

- For $(q, q', i) \in Q_p \setminus \{\text{Fail}, \text{Undef}\}$, $1 \leq i \leq l-1$ and $\sigma \in \Sigma$
 - if $h_2(q, \sigma) = h'_2(q', \sigma) \neq \perp$ then $H((q, q', i), \sigma) = ((h_1(q, \sigma), h'_1(q', \sigma), i+1), h_2(q, \sigma))$
 - if $h_2(q, \sigma) = h'_2(q', \sigma) = \perp$ then $H((q, q', i), \sigma) = (\text{Undef}, \text{undef})$
 - else $H((q, q', i), \sigma) = (\text{Fail}, \text{fail})$.
- For $(q, q', l) \in Q_p \setminus \{\text{Fail}, \text{Undef}\}$ and $\sigma \in \Sigma$, $H((q, q', l), \sigma) = \perp$.
- For $\sigma \in \Sigma$, $H(\text{Fail}, \sigma) = \perp$ and $H(\text{Undef}, \sigma) = \perp$.

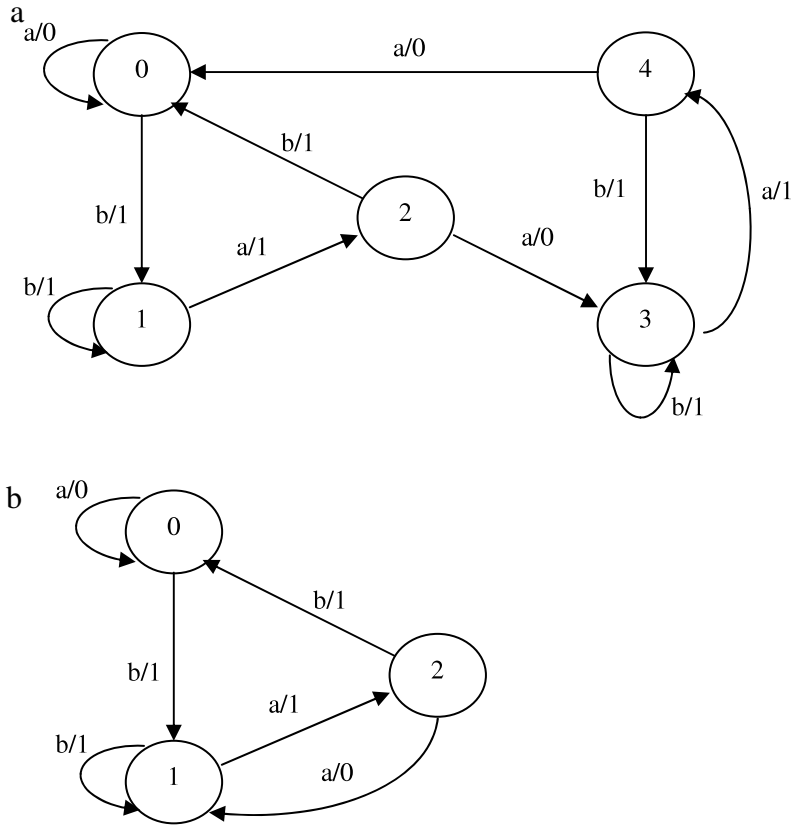


Fig. 5. Transition diagrams of N (a) and N' (b).

From the above construction it follows that checking that M and M' produce identical results for all input sequences of length less than or equal to l corresponds to establishing that the *Fail* state of $P_l(M, M')$ is not reachable. This is shown in the remainder of this section.

Lemma 9.1. Given $s \in \Sigma^*$, $q \in Q$, $q' \in Q'$ and $1 \leq i \leq l$, $H_1^*((q_0, q'_0, 0), s) = (q, q', i)$ if and only if $h_1^*(q_0, s) = q$, $h_1^*(q'_0, s) = q'$, $h_2^*(q_0, s) = h_2^*(q'_0, s)$ and $\|s\| = i$.

Proof. Follows from Definition 9.1 by induction on i . \square

Lemma 9.2. Given $s \in \Sigma^*$, s reaches *Fail* in $P_l(M, M')$ if and only if $s = s_1\sigma$ with $s_1 \in \Sigma[l-1]$ and $\sigma \in \Sigma$ such that $h_2^*(q_0, s_1) = h_2^*(q'_0, s_1) \neq \perp$ and $h_2^*(q_0, s_1\sigma) \neq h_2^*(q'_0, s_1\sigma)$ (this includes the case in which one of $h_2^*(q_0, s_1\sigma)$ and $h_2^*(q'_0, s_1\sigma)$ is defined and the other is not).

Proof. By Definition 9.1, s reaches *Fail* in $P_l(M, M')$ if and only if $s = s_1\sigma$ with $s_1 \in \Sigma[l-1]$ and $\sigma \in \Sigma$ for which there exist $q \in Q$, $q' \in Q'$ and i , $1 \leq i \leq l-1$, such that $H_1^*((q_0, q'_0, 0), s_1) = (q, q', i)$ and $H_1((q, q', i), \sigma) = \text{Fail}$. By Lemma 9.1, $H_1^*((q_0, q'_0, 0), s_1) = (q, q', i)$ if and only if $h_1^*(q_0, s_1) = q$, $h_1^*(q'_0, s_1) = q'$, $h_2^*(q_0, s_1) = h_2^*(q'_0, s_1)$ and $\|s_1\| = i$. By Definition 9.1, $H_1((q, q', i), \sigma) = \text{Fail}$ if and only if $h_2(q, \sigma) \neq h_2(q', \sigma)$. Thus, s reaches *Fail* in $P_l(M, M')$ if and only if $s = s_1\sigma$ with $s_1 \in \Sigma[l-1]$ and $\sigma \in \Sigma$ such that $h_2^*(q_0, s_1) = h_2^*(q'_0, s_1) \neq \perp$ and $h_2^*(q_0, s_1\sigma) \neq h_2^*(q'_0, s_1\sigma)$. \square

Lemma 9.3. *Fail* is not reachable in $P_l(M, M')$ if and only if M and M' are $\Sigma[l]$ -equivalent.

Proof. M and M' are $\Sigma[l]$ -distinguishable if and only if there exist $s_1 \in \Sigma[l-1]$ and $\sigma \in \Sigma$ such that $h_2^*(q_0, s_1) = h_2^*(q'_0, s_1) \neq \perp$ and $h_2^*(q_0, s_1\sigma) \neq h_2^*(q'_0, s_1\sigma)$. Then the result follows from Lemma 9.2. \square

10. Test suite validation through state-counting

Let M be the DFSM specification and M' the (unknown) DFSM model of the implementation under test. We need to establish that whenever M and M' behave identically for every test sequence, they will behave identically for every sequence of length less than or equal to the upper bound l . Then, using the above results, it remains to show that *Fail* is not reachable whenever M and M' produce identical results for every sequence in the test suite. As $Z_k \subseteq Y_k$, it is sufficient to prove the result for Z_k . In what follows M and M' are (possibly partially specified) DFSMs, M is l -minimal. m and m' denote the number of states of M and M' , respectively.

For simplicity, we assume that S contains exactly m elements; that is, for each $q \in Q$ there is a unique $s_q \in S$ such that s_q reaches q from q_0 and $\|s_q\| = \text{level}_M(q)$. Then, by Lemma 10.1 below, all sequences in S will have length at most $l - 1$.

Lemma 10.1. For every $q \in Q$, $\text{level}_M(q) \leq l - 1$.

Proof. If $\text{level}_M(q) \geq l$ for some $q \in Q$ then for every $p \in Q$, $p \sim_M^l q$. This contradicts the fact that M is l -reduced. \square

We use *state-counting* to show that, if *Fail* was reachable, Z_k would contain at least one “minimal” input sequence that would reach *Fail*. Among the shortest sequences, the minimal sequences are those sequences x for which also the “distance” $d(x, S)$ to the set S is the shortest. This is now defined.

Given $x \in \Sigma^*$ and $A \subseteq \Sigma^*$ with $\epsilon \in A$, the length of the shortest sequences $t \in \Sigma^*$ for which there exists a sequence $s \in A^*$ such that $st = x$ is denoted by $d(x, A)$, i.e. $d(x, A) = \min(\{\|t\| \mid t \in \Sigma^*, \exists s \in A^* \cdot st = x\})$. Since $\epsilon \in A$, the set $\{t \in \Sigma^* \mid \exists s \in A^* \cdot st = x\}$ is not empty, so $d(x, A)$ is well defined.

Now, consider a path through $P_l(M, M')$ formed by following a sequence t of length $k + 1$ after some sequence s from S . Then some prefix t will have reached a state of $P_l(M, M')$ for which an equivalent state (with respect to the sequences in $\Sigma[l]$) has been visited before by the path or has been reached by some sequence from S (Lemma 10.2). Consequently, if the sequences that reach the two equivalent states are denoted by y_1 and y_2 , respectively, y_2 can substitute y_1 in any path that would reach *Fail* (Lemma 10.3). By combining these two results, we show that Z_k would contain a “minimal” input sequence that would reach *Fail*, if this was reachable (Lemma 10.4).

Lemma 10.2. Suppose M' is l -minimal and $m' - m \leq k$. Let $s \in S$ and $t \in \Sigma^{k+1}$ such that $\|st\| \leq l$ and s is the longest prefix of st that is in S . Suppose $h_1^*(q_0, st) = q$, $q \in Q$ and let $\text{pref}(t)$ denote the set of all prefixes of t . If M and M' are $((S \cup \{s\}\text{pref}(t) \setminus \{st\})W_\epsilon \cup \{st\}W_q \cup \{st\}) \cap \Sigma[l]$ -equivalent then there exist $y_1 \in \{s\}\text{pref}(t) \setminus \{s\}$ and $y_2 \in S \cup \text{pref}(y_1) \setminus \{y_1\}$ such that the following two conditions hold:

- $\|y_2\| < \|y_1\|$ or $\|y_2\| \leq \|y_1\|$ and $d(y_2, S) < d(y_1, S)$
- $H_1^*((q_0, q'_0, 0), y_1) = (q_1, q', \|y_1\|)$ and $H_1^*((q_0, q'_0, 0), y_2) = (q_2, q', \|y_2\|)$ for some $\Sigma[l - \|y_1\|]$ -equivalent states $q_1, q_2 \in Q$ and some state $q' \in Q'$.

Proof. As all sequences in $S \cup \{s\}\text{pref}(t)$ have length at most l , M and M' are $(S \cup \{s\}\text{pref}(t))$ -equivalent. Then, by Lemma 9.1, for all $x \in S \cup \{s\}\text{pref}(t)$, $H_1^*((q_0, q'_0, 0), x) \in Q_p \setminus \{\text{Fail}, \text{Undef}\}$. Since M and M' are S -equivalent and M is l -reduced, the states reached by S in M' are pairwise l -dissimilar, so S reaches m states of M' . On the other hand, the set $\{s\}\text{pref}(t) \setminus \{s\}$ contains $k + 1$ elements, none of which is contained in S . Since $m + k + 1 > m'$, some sequence in $\{s\}\text{pref}(t) \setminus \{s\}$ will reach a state q' of M' that either has been met before by this sequence or has already been reached by a sequence in S . Thus there exist $y_1 \in \{s\}\text{pref}(t) \setminus \{s\}$ and $y_2 \in S \cup \{s\}\text{pref}(y_1) \setminus \{y_1\}$ such that $h_1^*(q'_0, y_1) = q'$ and $h_1^*(q'_0, y_2) = q'$. Let $h_1^*(q_0, y_1) = q_1$ and $h_1^*(q_0, y_2) = q_2$. Then $H_1^*((q_0, q'_0, 0), y_1) = (q_1, q', \|y_1\|)$ and $H_1^*((q_0, q'_0, 0), y_2) = (q_2, q', \|y_2\|)$.

Let $\mu = \max\{\|y_1\|, \|y_2\|\}$. We prove by contradiction that q_1 and q_2 are $\Sigma[l - \mu]$ -equivalent. Assume q_1 and q_2 are $\Sigma[l - \mu]$ -distinguishable, $\mu < l$. Since W_{q_1} is a strong identification set of q_1 , q_1 and q_2 are $(W_{q_1} \cap \Sigma[l - \mu])$ -distinguishable. Two cases can be distinguished:

- $y_1 \neq st$. Then M and M' are $(\{y_1, y_2\}W \cap \Sigma[l])$ -equivalent. Thus q_1 and q' are $(W \cap \Sigma[l - \|y_1\|])$ -equivalent and q_2 and q' are $(W \cap \Sigma[l - \|y_2\|])$ -equivalent. Hence q_1 and q_2 are $(W \cap \Sigma[l - \mu])$ -equivalent. As $W_{q_1} \subseteq W$, this provides a contradiction, as required.
- $y_1 = st$. Then $q_1 = q$ and M and M' are $(\{y_1\}W_{q_1} \cup \{y_2\}W) \cap \Sigma[l]$ -equivalent. Thus q_1 and q' are $(W_{q_1} \cap \Sigma[l - \|y_1\|])$ -equivalent and q_2 and q' are $(W \cap \Sigma[l - \|y_2\|])$ -equivalent. As $W_{q_1} \subseteq W$, q_1 and q_2 are $(W_{q_1} \cap \Sigma[l - \mu])$ -equivalent. This provides a contradiction, as required.

We now show that $\|y_2\| < \|y_1\|$ or $\|y_2\| \leq \|y_1\|$ and $d(y_2, S) < d(y_1, S)$. If $y_2 \in \text{pref}(y_1) \setminus \{y_1\}$, $\|y_2\| < \|y_1\|$. Otherwise $y_2 \in S \setminus \{s\}$, so $\|y_2\| = \text{level}_M(q_2)$. Then there are two cases:

- $q_1 = q_2$. Then $\text{level}_M(q_2) \leq \|y_1\|$ so $\|y_2\| \leq \|y_1\|$. Since $y_1 \notin S$ and $y_2 \in S$, $d(y_2, S) < d(y_1, S)$.
- $q_1 \neq q_2$. We prove by contradiction that $\|y_2\| < \|y_1\|$. Assume $\|y_1\| \leq \|y_2\|$. Then $\text{level}_M(q_1) \leq \|y_1\| \leq \|y_2\| = \text{level}_M(q_2)$. Hence $\text{level}_M(q_1) \leq \text{level}_M(q_2) = \|y_2\|$. As M is l -reduced, q_1 and q_2 are $\Sigma[l - \|y_2\|]$ -distinguishable. On the other hand, we have shown that q_1 and q_2 are $\Sigma[l - \mu]$ -equivalent. Since $\mu = \|y_2\|$, this is a contradiction.

Hence $\|y_2\| < \|y_1\|$ or $\|y_2\| \leq \|y_1\|$ and $d(y_2, S) < d(y_1, S)$. Since $\|y_2\| \leq \|y_1\|$, $\mu = \|y_1\|$, so q_1 and q_2 are $\Sigma[l - \|y_1\|]$ -equivalent. \square

Lemma 10.3. Let $(q_1, q', j_1), (q_2, q', j_2) \in Q_p \setminus \{\text{Fail}, \text{Undef}\}$, $0 \leq j_2 \leq j_1 \leq l - 1$, and $x \in \Sigma^*$, $1 \leq \|x\| \leq l - j_1$. Suppose q_1 and q_2 are $\Sigma[l - j_1]$ -equivalent states of M . If x reaches *Fail* from (q_1, q', j_1) then x reaches *Fail* from (q_2, q', j_2) .

Proof. If x reaches *Fail* from (q_1, q', j_1) then $x = s\sigma$ with $s \in \Sigma[l - j_1 - 1]$ and $\sigma \in \Sigma$ for which there exist $p_1 \in Q$, $p' \in Q'$ such that $H_1^*((q_1, q', j_1), s) = (p_1, p', j_1 + \|s\|)$ and $H_1((p_1, p', j_1 + \|s\|), \sigma) = \text{Fail}$. Since q_1 and q_2 are $\Sigma[l - j_1]$ -equivalent and $j_2 \leq j_1$, $H_1^*((q_2, q', j_2), s) = (p_2, p', j_2 + \|s\|)$ with $p_2 \in Q$ such that p_1 and p_2 are $\Sigma[l - j_1 - \|s\|]$ -equivalent. As $H_1((p_1, p', j_1 + \|s\|), \sigma) = \text{Fail}$ and $l - j_1 - \|s\| \geq 1$, it follows that $H_1((p_2, p', j_2 + \|s\|), \sigma) = \text{Fail}$. Thus x reaches *Fail* from (q_2, q', j_2) . \square

Lemma 10.4. *Suppose M' is l -minimal and $m' - m \leq k$. If M and M' are Z_k -equivalent then $Fail$ is not reachable in $P_l(M, M')$.*

Proof. We provide a proof by contradiction. Assume $Fail$ is reachable and let X be the set of all sequences of minimum length that reach $Fail$ from the initial state of $P_l(M, M')$. Let $\mu = \min\{d(x, S) \mid x \in X\}$ and $X_\mu = \{x \in X \mid d(x, S) = \mu\}$.

We prove by contradiction that $X_\mu \cap S\Sigma[k+1] \neq \emptyset$. Assume $X_\mu \cap S\Sigma[k+1] = \emptyset$ and let $x \in X_\mu$. Then $x \notin S\Sigma[k+1]$. Since $\epsilon \in S$, $x \in S\Sigma^*$. Let $s \in S$ be the longest prefix of x that is in S . Then $x = stu$, for some $t \in \Sigma^{k+1}$ and $u \in \Sigma^* \setminus \{\epsilon\}$ with $\|stu\| \leq l$. By Lemmas 9.2 and 9.1, $h_2^*(q_0, st) = h_2^*(q'_0, st) \neq \perp$. Let $h_1^*(q_0, st) = q$. Since M and M' are Z_k -equivalent, M and M' are $((S \cup \{s\}pref(t) \setminus \{st\})W_\epsilon \cup \{st\}W_q \cup \{st\}) \cap \Sigma[l]$ -equivalent. Then, by Lemma 10.2, there exist $y_1 \in \{s\}pref(t) \setminus \{s\}$ and $y_2 \in S \cup pref(y_1) \setminus \{y_1\}$ such that the following hold:

- $\|y_2\| < \|y_1\|$ or $\|y_2\| \leq \|y_1\|$ and $d(y_2, S) < d(y_1, S)$
- $H_1^*((q_0, q'_0, 0), y_1) = (q_1, q', \|y_1\|)$ and $H_1^*((q_0, q'_0, 0), y_2) = (q_2, q', \|y_2\|)$ for some $\Sigma[l - \|y_1\|]$ -equivalent states $q_1, q_2 \in Q$ and some state $q' \in Q'$.

Let $z \in \Sigma^*$ such that $st = y_1z$. As x reaches $Fail$ from $(q_0, q'_0, 0)$, zu reaches $Fail$ from $(q_1, q', \|y_1\|)$. Since q_1 and q_2 are $\Sigma[l - \|y_1\|]$ -equivalent states of Z , $\|zu\| \leq l - \|y_1\|$ and $\|y_2\| \leq \|y_1\|$, by Lemma 10.3, zu reaches $Fail$ from $(q_2, q', \|y_2\|)$. Thus, y_2zu reaches $Fail$ from $(q_0, q'_0, 0)$. If $\|y_2\| < \|y_1\|$ then y_2zu is a sequence shorter than x that reaches $Fail$ from the initial state of $P_l(M, M')$. Thus $x \notin X$, which is a contradiction. Otherwise, $\|y_2\| = \|y_1\|$ and $d(y_2, S) < d(y_1, S)$. Since no sequence in $\{y_1\}pref(zu)$ is contained in S , $d(y_2zu, S) < d(y_1zu, S)$. Consequently $\|y_2zu\| = \|x\|$ and $d(y_2zu, S) < d(x, S)$. Thus $x \notin X_\mu$, which provides a contradiction, as required. Thus $X_\mu \cap S\Sigma[k+1] \neq \emptyset$.

On the other hand, M and M' are $((S\Sigma[k] \cup R\Sigma[k]) \cap \Sigma[l])$ -equivalent. Since $R = S\Sigma \setminus S$, M and M' are $(S\Sigma[k+1] \cap \Sigma[l])$ -equivalent. Thus, by Lemma 9.2, no sequence in $S\Sigma[k+1]$ will reach $Fail$ from the initial state of $P_l(M, M')$. This provides a contradiction, as required. Hence $Fail$ is not reachable. \square

Theorem 10.1. *Suppose M' is l -minimal and $m' - m \leq k$. M and M' are $\Sigma[l]$ -equivalent if and only if M and M' are Z_k -equivalent.*

Proof. “ \Rightarrow ” Obvious, since $Z_k \subseteq \Sigma[l]$.

“ \Leftarrow ”: Follows from Lemmas 10.4 and 9.3. \square

Theorem 10.2. *Suppose $m' - m \leq k$. M and M' are $\Sigma[l]$ -equivalent if and only if M and M' are Z_k -equivalent.*

Proof. If M' is l -minimal then the result follows directly from Theorem 10.1. Otherwise, there exists a reachable and l -reduced DFSM M'' , $\Sigma[l]$ -equivalent to M' , for which the number of states m'' is less than m' . As $m'' - m \leq k$, by Lemma 10.4, M and M'' are $\Sigma[l]$ -equivalent if and only if M and M'' are Z_k -equivalent. Hence M and M' are $\Sigma[l]$ -equivalent if and only if M and M' are Z_k -equivalent. \square

Theorem 10.3. *Suppose $m' - m \leq k$. M and M' are $\Sigma[l]$ -equivalent if and only if M and M' are Y_k -equivalent.*

Proof. Follows from Theorem 10.2 since $Z_k \subseteq Y_k \subseteq \Sigma[l]$. \square

Test suites for the unbounded case can be obtained by removing the upper bound (i.e. considering l sufficiently large) from the formulae of Y_k and Z_k . These are exactly the test suites produced by the original methods (given in Sections 3.1 and 3.2), with the mention that S , W and \mathcal{W} are required to be a proper state cover, a strong characterization set and a strong identification set, respectively. Recall that, for completely specified DFSMs, the test suites for unbounded sequences were $U_k = S\Sigma[k+1]W$ and $V_k = S\Sigma[k]W \cup R\Sigma[k] \otimes \mathcal{W}$. These were extended to cope with (possibly) partially specified DFSMs; the extended formulae were $U'_k = S\Sigma[k+1]W_\epsilon \setminus \{\epsilon\}$ and $V'_k = S\Sigma[k]W_\epsilon \cup R\Sigma[k] \otimes \mathcal{W}_\epsilon \setminus \{\epsilon\}$, respectively. As $V_k \subseteq U_k$ and $V'_k \subseteq U'_k$ we only state the results for V_k and V'_k , respectively. Naturally, in the case of unbounded sequences, the specification M is assumed to be a minimal DFSM.

Corollary 10.1. *Suppose $m' - m \leq k$. M and M' are equivalent if and only if M and M' are V'_k -equivalent.*

Proof. “ \Rightarrow ” Obvious.

“ \Leftarrow ”: Assume M and M' are V'_k -equivalent but not equivalent. Then there exists an input sequence s that distinguishes between M and M' . Let l_1 be the length of this sequence. On the other hand, since M is minimal, there exists an integer $l_2 \geq 1$ such that M is l -minimal for every $l \geq l_2$. Let $l = \max\{l_1, l_2\}$. Then M is l -minimal and M and M' are $(V'_k \cap \Sigma[l])$ -equivalent. Thus, by Theorem 10.2, M and M' are $\Sigma[l]$ -equivalent. As $\|s\| \leq l$, s does not distinguish between M and M' . This provides a contradiction. \square

Corollary 10.2. *Suppose M and M' are completely specified and $m' - m \leq k$. M and M' are equivalent if and only if M and M' are V_k -equivalent.*

Proof. “ \Rightarrow ” Obvious.

“ \Leftarrow ”: If M and M' are V_k -equivalent then the result follows from Corollary 10.1. Otherwise, there exists $s \in V'_k \setminus V_k$ that distinguishes between M and M' . As $V'_k \setminus V_k \subseteq S\Sigma[k] \cup R\Sigma[k]$, every sequence in $V'_k \setminus V_k$ is a prefix of some sequence in V_k . Thus, s is a prefix of some sequence $t \in V_k$. As M and M' are completely specified, t also distinguishes between M and M' . Thus, M and M' are V_k -distinguishable, which contradicts the original hypothesis. \square

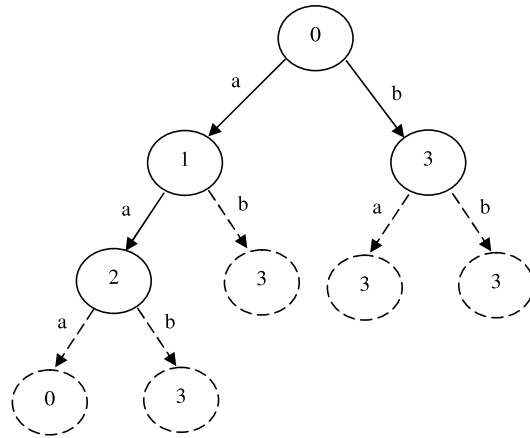


Fig. 6. Testing tree for M and $n = 2$.

11. Implementation and complexity

A state cover S of M can be obtained by constructing a testing tree T in a breadth-first fashion, as follows [5]:

- Label the root of T with the initial state of M . This is the level 0 of M .
- Suppose we have already built T to a level i . Then the $(i + 1)$ th level is built by examining nodes at the i th level from left to right. A node at the i th level is terminal if its label is the same as a non-terminal node at some level j , $j \leq i$. Otherwise, a branch is attached for each transition emerging from the state which labels the node.

Since each level contains at least one non-terminal node, the tree will have at most m levels, where m denotes the number of states of M . For M as represented in Fig. 1(a) and $n = 2$, the testing tree is given in Fig. 6; the terminal nodes and the arcs leading to them are in dashed line. A state cover S can be then constructed by enumerating all partial paths of T that lead to non-terminal states. (In fact, it is more convenient to directly construct a transition cover $P = S \cup S\Sigma$ of M by enumerating all partial paths of T .) As T contains $m \cdot r$ arcs and $m \cdot r + 1$ partial paths, where r represents the size of the input set, the maximum amount of work required to generate S or P will be proportional to $m \cdot r$. Also note that for each state q , a path of minimum length that reaches q will be selected, so the set S thus constructed will be a proper state cover.

A characterization set W can be obtained by constructing a sequence of equivalence relations $\equiv_1, \dots, \equiv_j$ on the state set, where $q \equiv_i q'$ if and only if the outputs produced from q and q' by every sequence of length less than or equal to i are identical. As the sequences in $\Sigma[i + 1]$ will distinguish at least one more pair of states than the sequences in $\Sigma[i]$, there will be at most $m - 1$ such equivalence relations, i.e. $j \leq m - 1$. The relations \equiv_i can be constructed from the so-called P_i tables (see [10]). The amount of work required to construct a P_i table is proportional to $m \cdot r$, the number of entries in the table. Thus the maximum amount of work required to generate W will be proportional to $(m - 1) \cdot m \cdot r$ or, roughly, to $m^2 \cdot r$. Analogously to the construction of S , the algorithm will select the shortest possible sequences, so the set W thus constructed will be a strong characterization set. Consequently, the complexity of the test generation algorithm for the bounded case is the same as for the unbounded case.

According to [5], the upper bound for the number of sequences in $S\Sigma[k + 1]W$ is $m^2 \cdot r^{k+1}$ and the total length of all sequences is not greater than $m^2 \cdot m' \cdot r^{k+1}$, where m' is the (estimated) maximum number of states of M' . In particular, for $k = 0$, the respective bounds are $m^2 \cdot r$ and $m^3 \cdot r$. Note that these bounds refer to the worst case; in an average case, the size of the test suite is much lower. The increase in size produced by replacing W with W_ϵ in the above formula is negligible. (Indeed, the upper bounds for the number of extra sequences and the total length of extra sequences are proportional to mr^{k+1} and m^2r^{k+1} , respectively. Since $m \leq m'$, these figures are negligible compared to the original bounds, m^2r^{k+1} and $m^2m'r^{k+1}$, respectively.) On the other hand, by removing the sequences of length more than l , the size of the test suite may be considerably reduced. For our example (M in Fig. 1(a) and $l = m - 1$, where $m = n + 2$), a (proper) state cover and a (strong) characterization set produced by the aforementioned algorithms are $S = \{\epsilon, \dots, a^{m-2}, b\}$ and $W = \{a, \dots, a^{m-2}, b\}$, respectively. Then the number of sequences in $S\Sigma[1]W_\epsilon \cap \Sigma[m - 1] = \{a, \dots, a^{m-1}\} \cup \{a^i b a^j \mid 0 \leq i \leq m - 2, 0 \leq j \leq m - 2 - i\} \cup \{a^i b b \mid 0 \leq i \leq m - 3\} \cup \{b b a^i \mid 1 \leq i \leq m - 3\} \cup \{b a b, b b b\}$ is approximately $\sum_{i=1}^{m-1} i \approx m^2/2$ and the total length of these sequences is approximately $\sum_{i=1}^{m-1} i^2 \approx m^3/3$. The above figures represent approximately half of the number of sequences in $S\Sigma[1]W = \{a, \dots, a^{2-m-3}\} \cup \{a^i b a^j \mid 0 \leq i \leq m - 2, 0 \leq j \leq m - 2\} \cup \{a^{m-1} b, b a^{m-1}\} \cup \{a^i b b \mid 0 \leq i \leq m - 2\} \cup \{b b a^i \mid 1 \leq i \leq m - 2\} \cup \{b a b, b b b\}$ and one third of their total length, respectively.

Furthermore, the size (the number of states m in the above formulae) of the l -minimal DFSM used as basis for test generation may be significant lower than the size of the minimal DFSM. (Recall that, as shown in Section 5, every l -minimal DFSM is a minimal DFSM but not vice versa). For example, the DFSM represented in Fig. 7 is $\Sigma[n + 1]$ -equivalent to M and produces 0s for the remaining transitions (i.e. which correspond to sequences longer than $n + 1$). This DFSM is minimal and has $2 \cdot n + 2$ states whereas the $(n + 1)$ -minimal DFSM had only $n + 2$ states.

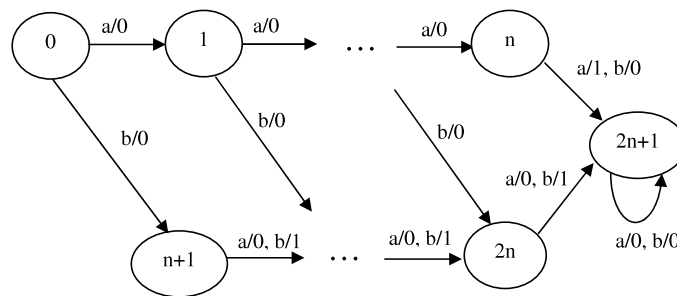


Fig. 7. Transition diagram of minimal DFSM.

12. Conclusions

This paper extends the W - and Wp -methods to the case of bounded sequences. The new methods may produce considerably smaller test suites than the originals while the complexity of the test generation algorithms remains basically unchanged. Furthermore, the methods for bounded sequences are stronger than the originals, as test suites for the unbounded case can be directly derived from the formulae for the bounded case.

A paper in progress considers bounded sequence test selection from *non-deterministic* FSMs. Possible future work also involves the generalization of these bounded sequence testing methods to classes of extended finite state machines, such as stream X-machines [13], using techniques similar to those employed in the unbounded case [16–18].

Acknowledgements

This research is supported by CNCSIS grant IDEI no. 496/2009, *An integrated evolutionary approach to formal modelling and testing* (EvoMT). The author would like to thank the anonymous reviewers, whose comments have improved the presentation of this paper.

References

- [1] T. Bălănescu, M. Gheorghe, F. Ipate, M. Holcombe, Formal black box testing for partially specified deterministic finite state machines, *Foundations of Computing and Decision Systems* 28 (1) (2003) 17–28.
- [2] T. Bălănescu, F. Ipate, The Wp method for partially specified deterministic finite state machines, *Annals of Bucharest University: Computer Science* LIII (1) (2004) 47–60.
- [3] A. Biere, A. Cimatti, E.M. Clarke, Y. Zhu, Symbolic Model Checking without BDDs, in: *Proceedings of Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS, 1999*, pp. 193–207.
- [4] C. Campeanu, N. Santean, S. Yu, Minimal cover automata for finite languages, *Theoretical Computer Science* 267 (1999) 3–16.
- [5] T.S. Chow, Testing software design modeled by finite state machines, *IEEE Transactions on Software Engineering* 4 (3) (1978) 178–187.
- [6] S. Eilenberg, *Automata, languages and machines*, vol. A., Academic Press, New York, 1994.
- [7] A. En-Nouaary, R. Dssouli, F. Khendek, Timed Wp -method: Testing real-time systems, *IEEE Transactions on Software Engineering* 28 (11) (2002) 1023–1038.
- [8] T. Filkorn, A method for symbolic verification of synchronous circuits, in: *Proc. Int. Symp. Comput. Hardware Description Lang. Applicat.*, 1991, pp. 249–259.
- [9] S. Fujiwara, G. von Bochmann, F. Khendek, M. Amalou, A. Ghedamsi, Test selection based on finite state models, *IEEE Transactions on Software Engineering* 17 (6) (1991) 591–603.
- [10] A. Gill, *Introduction to the Theory of Finite-State Machines*, McGraw-Hill, 1962.
- [11] D. Harel, M. Politi, *Modeling Reactive Systems with Statecharts: the STATEMATE Approach*, McGraw-Hill, New York, 1998.
- [12] R.M. Hierons, Testing from a non-deterministic finite state machine using adaptive state counting, *IEEE Transactions on Computers* 53 (10) (2004) 1330–1342.
- [13] M. Holcombe, F. Ipate, *Correct Systems: Building a Business Process Solution*, Springer Verlag, Berlin, 1998.
- [14] J.E. Hopcroft, J.D. Ullman, *Introduction to Automata Theory, Languages and Computation.*, Addison Wesley, Reading, MA, 1979.
- [15] F. Ipate, On the minimality of finite automata and stream x-machines for finite languages, *The Computer Journal* 48 (2) (2005) 157–167.
- [16] F. Ipate, M. Holcombe, An integration testing method that is proved to find all faults, *International Journal of Computer Mathematics* 63 (1997) 159–178.
- [17] F. Ipate, Complete deterministic stream X-machine testing, *Formal Aspects of Computing* 16 (4) (2004) 374–386.
- [18] F. Ipate, Testing against a non-controllable stream X-machine using state counting, *Theoretical Computer Science* 353 (1–3) (2006) 291–316.
- [19] ITU-T. Recommendation Z.100 Specification and description language (SDL). International Telecommunications Union, Geneva, Switzerland, 1999.
- [20] J.-H.R. Jiang, R.K. Brayton, On the verification of sequential equivalence, *International Workshop on Logic and Synthesis, (IWLS)*, 22 (6) (2002) 307–314.
- [21] H. Korner, On minimizing cover automata for finite languages in $O(n \log n)$ time, *LNCS 2608* (2003) 117–127.
- [22] D. Lee, M. Yannakakis, Principles and methods of testing finite state machines – a survey, *Proceedings of the IEEE* 84 (8) (1996) 1090–1123.
- [23] A. Miczo, *Digital Logic Testing and Simulation*, 2nd ed., John Wiley & Sons, 2003.
- [24] A. Paun, N. Santean, S. Yu, An $O(n^2)$ algorithm for constructing minimal cover automata for finite languages, in: *Lecture Notes in Computer Science*, vol. 2088, Springer, 2001, pp. 243–251.
- [25] A. Petrenko, N. Yevtushenko, G. von Bochmann, Testing deterministic implementations from nondeterministic FSM specifications, in: *Proc. of 9th International Workshop on Testing of Communicating Systems, IWTC'S'96*, 1996, pp. 125–140.
- [26] A. Petrenko, N. Yevtushenko, Testing from partial deterministic FSM specifications, *IEEE Transactions on Computers* 54 (9) (2005) 1154–1165.
- [27] M.K. Prasad, A. Biere, A. Gupta, A survey of recent advances in SAT-based formal verification, *Software Tools for Technology Transfer, (STTT)*, 7 (2) (2005) 156–173.

- [28] T. Ramalingam, A. Das, K. Thulasiram, Fault detection and diagnosis capabilities of test sequence selection methods based on the fsm model, *Computer Communications* 18 (2) (1995) 113–122.
- [29] T. Ramalingam, A. Das, K. Thulasiram, On testing and diagnosis of communication protocols based on the fsm model, *Computer communications* 18 (5) (1995) 329–337.
- [30] Y.-N. Shen, F. Lombardi, A.T. Dahbura, Protocol conformance testing using multiple uio sequences, in: E. Brinksma, G. Scollo, C.A. Vissers (Eds.), *Protocol Specification, Testing and Verification IX*, Elsevier Science Publishers B.V, North-Holland, 1990, pp. 131–143.
- [31] D. Sidhu, T. Leung, Formal methods for protocol testing: A detailed study, *IEEE Transactions on Software Engineering* 15 (4) (1989) 413–426.
- [32] H. Ural, Formal methods for test sequence generation, *Computer Communications* 15 (5) (1992) 311–325.