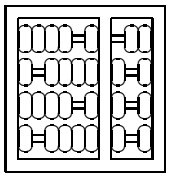

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Generalized Model-based Test Generation

A. L. Bonifácio *A. V. Moura* *A. S. Simão*

Technical Report - IC-08-014 - Relatório Técnico

May - 2008 - Maio

A Generalized Model-based Test Generation Method

Adilson Luiz Bonifácio* Arnaldo Vieira Moura† Adenilso da Silva Simão‡

Abstract

In this paper we present a generalization for the W-method [Cho78], which can be used for automatically generating test cases. In contrast to the W-method, this generalization allows for test case generations even in the absence of a characterization set for the specification. The work presents proofs of correctness for this generalization, and derives the original W-method from it as a particular case. Formal proofs of correctness for the W-method, not given in the original paper, are also presented in a clear and detailed way.

1 Introduction

In the literature, there are many model-based test derivation methods for conformance testing of critical and reactive systems [NS01, Tre99, CR05, Mye79]. Conformance testing aims at demonstrating that the implementation behavior conforms to the behavior dictated by the specification [Tre96, SkL89, Gar05]. The problem of generating test cases for conformance testing has been intensively studied, specially for models based on Finite State Machines (FSMs) [DEFY05, SL88, Gon70, Hen64]. One of the most well-known of these test generation methods is the W-method [Cho78], which uses the notion of characterization sets. The W-method was proposed for deterministic FSMs and it has been widely investigated [RU95, Hie06, Kri05], and many variations have been developed around its main ideas [LvBP94]. The partial W-method, the so called W_p-method [FBK$^+$91], is an extension to the W-method, where specific characterization subsets are associated with each machine state. Such modification potentially reduces the length of test sequences. The HSI-method [PvB95] was proposed to be used with partially specified FSMs. In this method, instead of a whole characterization set, a set of harmonized state identifiers is used for state identification and transition checking.

In this paper we present a generalization of the W-method. This generalization allows us to derive an m-complete test suite without using a characterization set, while considering deterministic FSM implementations with up to m states. In fact, our method can generate

*Computing Institute, University of Campinas, P.O. 6176 – Campinas – Brazil – 13081-970. Research supported in part by CNPq — Conselho Nacional de Desenvolvimento Científico e Tecnológico, grant #141978/2008-2

†Computing Institute, University of Campinas, P.O. 6176 – Campinas – Brazil – 13081-970.

‡Mathematic Science and Computing Institute, University of São Paulo, P.O. 668 – São Carlos – Brazil – 13560-970

test suites using only subsets of any characterization set. We discuss how to refine from the generalization in order to arrive at the original W-method, demonstrating that the latter is a particular case for our method. Proofs of correctness are presented in a clear form, including the correctness for the original W-method.

This paper is organized as follows. In Section 2 we describe some work related to our proposal, and we review some basic concepts. The concept of transition covers for FSMs is presented in Section 3. In Section 4 we introduce equivalence in FSMs, and stratified families of sets. The generation of a complete test suite is presented in Section 5. In Section 6 we reconsider characterization sets. How to refine our method in order to obtain the original W-method is in Section 7. In Section 8 we present the algorithm for the generalized method, and illustrate its usefulness with an example in Section 9. Finally, in Section 10, we give some concluding remarks.

2 Related Works

This section reviews the FSM model and some important related notions. We also present more details about the W-method and other variant model-based test generation methods, such as W_p and HSI.

2.1 Finite State Machines

The model used to capture systems' behaviors is the FSM. Formally, an FSM [Gil62] is a system $M = (X, Y, S, s_0, \delta, \lambda)$ given by:

- a finite input alphabet, X;
- a finite output alphabet, Y;
- a finite set of states, S;
- an initial state $s_0 \in S$; and
- output and transition functions, respectively, $\lambda : X \times S \to Y$ and $\delta : X \times S \to S$.

Note that such a machine is deterministic and complete. An FSM is called complete if for each state s of M, there is a transition from s, with input symbol a, for every $a \in X$. A deterministic FSM does not allow two different transitions going out of the same state with identical input symbols.

Successive applications of the transition function δ give ride to the extended transition function $\hat{\delta} : X^\star \times S \to S$, defined by

$$\begin{cases} \hat{\delta}(\epsilon, s) &= s, \\ \hat{\delta}(a\rho, s) &= \hat{\delta}(\rho, \delta(a, s)), \text{ where } a \in X \text{ and } \rho \in X^\star. \end{cases}$$

For convenience, if $\hat{\delta}(\rho, s_1) = s_2$, we also write $s_1 \xrightarrow{\rho} s_2$.

We extend λ to $\hat{\lambda} : X^\star \times S \rightarrow Y^\star$ in the form

$$\begin{cases} \hat{\lambda}(\epsilon, s) &= \epsilon, \\ \hat{\lambda}(a\rho, s) &= \lambda(a, s)\hat{\lambda}(\rho, \delta(a, s)), \text{where } a \in X \text{ and } \rho \in X^\star. \end{cases}$$

Henceforth, unless mention to the contrary, we will assume that M and M' denote FSMs in the form $M = (X, Y, S, s_0, \delta, \lambda)$ and $M' = (X, Y', S', s'_0, \delta', \lambda')$. Note that M and M' have the same input alphabet.

The reachability notion expresses the idea of starting at the initial state, traversing some transitions, and reaching a target state.

Definition 1 *A state s in a FSM M is reachable if and only if there exists $\rho \in X^\star$ such that $\widehat{\delta}(\rho, s_0) = s$.* ∎

We also say that $\hat{\lambda}(\rho, s)$ is the behavior of M from state s over the input sequence ρ. The behavior of M over ρ is simply the behavior of M from s_0 over ρ. A sequence ρ distinguishes two states s_1 and s_2 of M if ρ gives distinct behaviors for s_1 and s_2, that is, if $\hat{\lambda}(\rho, s_1) \neq \hat{\lambda}(\rho, s_2)$.

2.2 FSM-based testing

Now, we describe the basic W-method, which can be used for test generation using FSM models. We also briefly describe the related W_p and the HSI methods.

2.2.1 The W-method

The objective in [Cho78] was to verify whether implementation models conformed to a specification model, as characterized by the behavior responses generated by external stimuli.

Basically, the application of this method consists in two main steps, given a specification FSM M and an implementation FSM M':

1. test sequences generation, based on M;

2. application of each test sequence to M and M', followed by comparing their respective behaviors.

The technique uses characterization sets of M in order to obtain a complete set of test case sequences. A characterization set, loosely speaking, can distinguish every pair of machine states. Let W be a characterization set for M. In order to obtain the test sequences, the W-method prefixes the sequences in W with certain sequences from $X^\star$, thus obtaining a set Z with such extended sequences from W. Furthermore, the method also computes a cover set P for M. Basically, from P one can traverse any edge of M. The desired set of test sequences is then given by the product PZ. More details to be presented in the sequel.

2.2.2 The W_p-method

A related technique, the so called W_p-method [FBK$^+$91], can potentially reduce the total length of the test sequences generated by the basic W-method. Again, let W be a characterization set for the specification model, M. For each state s_i of M, an identification subset $W_i \subseteq W$ is obtained. The idea is that for each state s_j of M, with $s_i \neq s_j$, there exists an input sequence $\rho_j \in W_i$ such that s_i and s_j are distinguishable by ρ, and no other proper subset of W_i has this property.

Then, a checking sequence for each state is prefixed to the corresponding identification set. A checking sequence for a given state is simply an input sequence reaching that state, when starting at the initial state.

In [FBK$^+$91], it is argued that the length of the resulting test sequences may be shorter, compared to those sequences obtained using the complete PZ concatenation of the basic W-method.

2.2.3 The HSI-method

The HSI-method [PvB95] uses the notion of trace-inclusion and a quasi-equivalence relation to verify conformance between partial non-deterministic FSM implementations and a given FSM specification. For that, so called harmonized sate identification sets are used instead of the identification subsets used in the W_p-method. Whereas identification sets fixed the sequences associated with a specific state s_i, a harmonized state identification sets, D_i, is also constructed by taking prefixes of a characterization set W, but now allowing the reuse of a same prefix for different states. Distinguishing sequences for states are then taken from the intersection of D_i-sets. The discussion in [PvB95] affirms that shorter sequences can be found to distinguish every pair of states in M.

3 Transition Covers

Let M be an FSM. A set $P \subseteq X^\star$ is required to exercise every transition in M, *i.e.*, for every transition $\delta(a, s) = r$ in M there must be $\rho, \rho a \in P$ such that $\hat{\delta}(\rho, s_0) = s$. In this way, we can obtain a behavior of M reaching state s, and terminating by traversing the specific edge from s to r labeled by a.

The cover notion is formalized next.

Definition 2 *A set of input sequences $C \subseteq X^\star$ is a cover for an FSM M if for every pair of states $s, r \in S$ and every input symbol $a \in X$, with $\delta(a, s) = r$, there exist $\rho \in C$ and $\rho a \in C$ such that $\widehat{\delta}(\rho, s_0) = s$.* ∎

The cover set P can be obtained by constructing a labeled tree for M. A labeled tree is a system $T = (N, A, l_v, l_e)$, where N is a set of nodes, A is the set of edges, and $l_v : N \to S$ and $l_e : N \times N \to X$ are labeling functions of nodes and edges, respectively. The nodes in the tree will be labeled by states of M and edges will be labeled by symbols from X.

Construction 3 *A labeled tree for M, $T = (N, A, l_v, l_e)$, is constructed as follows:*

1. *Initiate with $N = \{n_0\}$, $A = \emptyset$, $l_v(n_0) = s_0$ and $l_e = \emptyset$, where s_0 is the initial state of M and n_0 is the root of T. We say that n_0 has level zero in T.*

2. *Inductively, suppose T is already constructed up to level $k \geq 0$. Level $k+1$ is constructed by inspecting of nodes in level k from left to right:*

   (a) *let $n \in N$ be the next node to be inspected.*

   (b) *if there already exists $m \in N$ with $l_v(m) = l_v(n)$, and m is at some level $l < k$ in T, then node n is ignored, and we take the next node at level k. Otherwise, for every input $a \in X$ and every $r \in S$ with $r = \delta(a, l_v(n))$, we add a new node n' to N, a new edge (n, n') to A, and define $l_v(n') = r$ and $l_e(n, n') = a$. We then proceed to the next node in level k.*

3. *Step 2 is repeated if no new nodes were added to T; otherwise, T is completed.* ∎

The process will always terminate since the set of states in M is finite. Depending on how symbols of X are selected, different trees can be obtained (see step 2b in Construction 3).

The next lemma expresses a simple fact.

Lemma 4 *Let $T = (N, A, l_v, l_e)$ be a labeled tree for an FSM M. Let $n \in N$ and let α be the sequence of edge labels in the simple path from the root to n in T. Then, $\widehat{\delta}(\alpha, s_0) = l_v(n)$.*

Proof By a simple induction on the level of n in T. ∎

The next definition shows how to construct a required cover set.

Definition 5 *Let T be a labeled tree for M. The set P_T is defined by all words $\alpha \in X^\star$ which label simple paths in T, starting at the root and finishing at any node of T.* ∎

Note that $\epsilon \in P_T$. When T is clear from the context, we will use the simplified notation P instead of P_T.

We can now show that P_T, from Definition 5, is a cover set for machine M. Before that, we describe some properties of the labelled tree.

Lemma 6 *Let $T = (N, A, l_v, l_e)$ be a labeled tree for an FSM M, as given by Construction 3. Let P_T be the set obtained from the Definition 5. Let $\rho \in X^\star$ and $s \in S$ such that $\widehat{\delta}(\rho, s_0) = s$. Then, there exists a node $n \in N$ with $l_v(n) = s$. Furthermore, there exists a sequence $\alpha \in P_T$ with $\widehat{\delta}(\alpha, s_0) = s$ and such that for every edge $\delta(a, s) = r$ we have $\alpha a \in P_T$.*

Proof By induction on $|\rho| = 0$. If $|\rho| = 0$, then $\rho = \epsilon$ and $s = s_0$. Let n be the root of T. Then $l_v(n) = s_0 = s$, by step (1) of Construction 3. Now, choosing $\alpha = \epsilon$, we have $\alpha \in P_T$, with $\widehat{\delta}(\alpha, s_0) = s_0 = s$. Moreover, if $\delta(a, s) = r$ is an edge of M, then step (2b) of Construction 3 will add a node m to N, at level 1, and an edge (n, m) to A, labeling them $l_v(m) = r$ and $l_e(n, m) = a$. This comes from the fact that n is at the level zero in T (step (1) of Construction 3) and, obviously, there is no node at an inferior level. Clearly,

the sequence of edge labels in the path from the root n up to m in T is a. Then, $a \in P_T$. Hence, $\alpha a \in P_T$, because $\alpha a = a$.

Assume such result holds for every ρ with $|\rho| \leq k$, where $k \geq 0$. Let $\rho = \sigma b$, with $\sigma \in X^k$, $b \in X$ and $\widehat{\delta}(\rho, s_0) = s$. We have some $s_1 \in S$ such that $\widehat{\delta}(\sigma, s_0) = s_1$ and $\delta(b, s_1) = s$. By induction, there is a node $n \in N$ with $l_v(n) = s_1$. Let $h \geq 0$ be the level of n in T. Without loss of generality, we can assume that h is minimal. In this way, when n is examined during the step 2 (Construction 3), by using the minimality of h, given the edge $\delta(b, s_1) = s$ and knowing that $l_v(n) = s_1$, a new node n' will be added to N, at level $h+1$, with $l_v(n') = s$. Therefore, n' satisfies the first assertive of the lemma . Now we know there is a node in N whose label is s. Again, let $h' \geq 0$ be the lowest level of a node $m \in N$ such that $l_v(m) = s$. Let α be a sequence of edge labels in the path from the root up to m in T. From Lemma 4 we have $\widehat{\delta}(\alpha, s_0) = l_v(m) = s$ and, from the construction of P_T we have $\alpha \in P_T$. Consider an edge $\delta(a, s) = r$. The minimality of h' guarantees that step (2b) of Construction 3 will add a new node m' to N, at level $h' + 1$, and an edge (m, m') to A, with $l_e(m, m') = a$. By construction of P_T, $\alpha a \in P_T$, thus proving the second assertive of the lemma, and completing the proof. ∎

Now we can enunciate the cover property.

Corollary 7 *Let $T = (N, A, l_v, l_e)$ be the labeled tree for an FSM M, as given by Construction 3. Let P_T be the set obtained as in Definition 5. If every state of M is reachable, then the set $P_T \subseteq X^\star$ is a cover for M.*

Proof Let $\delta(a, r) = s$ be an edge. As r is reachable, we have $\widehat{\delta}(\rho, s_0) = r$, for some $\rho \in X^\star$. By Lemma 6, we have α, $\alpha a \in P_T$ with $\widehat{\delta}(\alpha, s_0) = r$. Hence, P_T is a cover for M. ∎

4 Equivalences and Stratified Families of Sets

This section deals with state equivalence relations, induced by the transition functions of the extended machines. The next definition exposes those notions in a general context.

Definition 8 *Let M and M' be two FSMs over the same input alphabet, X, and let s and s' be states of M and M', respectively.*

1. *Let $\rho \in X^\star$. We say s is ρ-equivalent to s' if $\widehat{\lambda}(\rho, s) = \widehat{\lambda'}(\rho, s')$. In this case, we write $s \approx_\rho s'$. Otherwise, s and s' are ρ-distinguishable and we write $s \not\approx_\rho s'$.*

2. *Let $K \subseteq X^\star$. We say s is K-equivalent to s' if s is ρ-equivalent to s', for every $\rho \in K$. In this case, we write $s \approx_K s'$. Otherwise, s and s' are K-distinguishable and we write $s \not\approx_K s'$.*

3. *Let $k \geq 0$. We say s is k-equivalent to s' if s is X^k-equivalent to s'. Otherwise, s and s' are k-distinguishable. We write, respectively, $s \approx_k s'$ and $s \not\approx_k s'$.*

4. *We say s is equivalent to s' if s is k-equivalent to s', for every $k \geq 0$. Otherwise, s and s' are distinguishable. We write, respectively, $s \approx s'$ and $s \not\approx s'$.*

We will avoid overloading the notation by indicating M and M' explicitly, e.g., in the form $\approx_k^{M,M'}$, since both machines will always be clear from the context. Definition 8, obviously, also applies when M and M' are the same machine. In this case, it is easy to verify that all defined relations above are, in fact, equivalence relations over the state set of the machine.

Definition 9 *Let M be an FSM. The index of M, ι_M, is the number of equivalence classes induced by the $\approx$ relation over the states of M.*

Clearly, we will always have $1 \leq \iota_M \leq |S|$, where S is the state set of M.

The next lemma gathers some simple observations.

Lemma 10 *Let M and M' be two FSMs with states s and s', respectively.*

1. *Let $K \subseteq X^\star$. If $s \approx_K s'$, then $s \approx_L s'$, for every L with $L \subseteq K$. On the other hand, if $s \not\approx_K s'$, then $s \not\approx_L s'$, for every L with $K \subseteq L$.*
2. *Let $k \geq 0$. If $s \approx_k s'$ then $s \approx_l s'$ for every l with $l \leq k$. On the other hand, if $s \not\approx_k s'$, then $s \not\approx_l s'$, for every l with $l \geq k$.*
3. *Let $K \subseteq X^\star$. If $s \not\approx_K s'$, then $s \not\approx_{KL} s'$, for every $L \neq \emptyset$.*

Proof Trivial. ∎

In the sequel, we will be considering specific sets of input sequences.

Definition 11 *Let $Z_i \subseteq X^\star$, $i \geq 0$, where X is an alphabet. We say that $\{Z_i\}_{i\geq 0}$ is a stratified family over X if*

1. *$Z_0 \neq \emptyset$; and*
2. *$(X \cup \{\epsilon\})Z_i = Z_{i+1}$, for every $i \geq 0$.*

It is easy to see that these properties are independent of each other.

Another characterization for stratification is given as follows.

Proposition 12 *Let $Z_i \subseteq X^\star$, $i \geq 0$, where X is an alphabet and with $Z_0 \neq \emptyset$. Then, the family $\{Z_i\}_{i\geq 0}$ is stratified if and only if $Z_k = \bigcup_{j=0}^{k} X^j Z_0$ for every $k \geq 0$.*

Proof Assume that $Z_k = \bigcup_{j=0}^{k} X^j Z_0$, for every $k \geq 0$. Then,

$$\begin{aligned}
(X \cup \{\epsilon\})Z_k &= XZ_k \bigcup Z_k \\
&= X\Big(\bigcup_{0\leq j\leq k} X^j Z_0\Big) \bigcup \Big(\bigcup_{0\leq j\leq k} X^j Z_0\Big) \\
&= \Big(\bigcup_{1\leq j\leq k+1} X^j Z_0\Big) \bigcup \Big(\bigcup_{0\leq j\leq k} X^j Z_0\Big) \\
&= \bigcup_{0\leq j\leq k+1} X^j Z_0 = Z_{k+1}.
\end{aligned}$$

Since we have $Z_0 \neq \emptyset$, the stratification is established.

Now assume that $\{Z_i\}_{i\geq 0}$ is a stratified family. When $k = 0$ is immediate that $Z_k = \bigcup_{j=0}^{k} X^j Z_0$. Continuing by induction, we assume that the result holds for $k-1$, where $k \geq 1$, and show that $Z_k = \bigcup_{j=0}^{k} X^j Z_0$.

Let $z \in Z_k$. Then, from Definition 11, $z \in (X \cup \{\epsilon\})Z_{k-1}$. If $z \in Z_{k-1}$ then the induction hypothesis guarantees that $z \in \bigcup_{j=0}^{k-1} X^j Z_0$, and then $z \in \bigcup_{j=0}^{k} X^j Z_0$. On the other hand, if $z \in XZ_{k-1}$, then $z = aw$, with $a \in X$ and $w \in Z_{k-1}$. From the induction hypothesis, $w \in \bigcup_{j=0}^{k-1} X^j Z_0$, and then $z \in \bigcup_{j=1}^{k} X^j Z_0$, and so, $z \in \bigcup_{j=0}^{k} X^j Z_0$. Therefore, $Z_k \subseteq \bigcup_{j=0}^{k} X^j Z_0$. Now let $z \in \bigcup_{j=0}^{k} X^j Z_0$. If $z \in \bigcup_{j=0}^{k-1} X^j Z_0$, the induction hypothesis gives $z \in Z_{k-1}$. From Definition 11(2), we obtain $z \in Z_k$. If $z \in X^k Z_0$, then $z = aw$ with $a \in X$ and $w \in X^{k-1}Z_0$. From the induction hypothesis, $w \in Z_{k-1}$ and, consequently, $z \in XZ_{k-1}$. Again, from Definition 11(2) we obtain $z \in Z_k$. Therefore, $\cup_{j=0}^{k} X^j Z_0 \subseteq Z_k$. We conclude that $Z_k = \cup_{j=0}^{k} X^j Z_0$, extending the induction. ∎

The next result guarantees that certain sequences always have continuations in some of the Z_k sets.

Lemma 13 *Let $\{Z_i\}_{i\geq 0}$ be a stratified family over X and let $k \geq 0$. Then*

1. *$Z_k \subseteq Z_j$, for every $j \geq k$; and*
2. *For every $\alpha \in X^j$, with $0 \leq j \leq k$, there exists $\beta \in X^\star$ such that $\alpha\beta \in Z_k$.*

Proof From Proposition 12, we deduce $Z_i \subseteq Z_{i+1}$, for every $i \geq 0$. A simple induction establishes item (1). For item (2), since $Z_0 \neq \emptyset$, we take $\gamma \in Z_0$. Since $j \leq k$, we take $\sigma \in X^{k-j}$. Hence, $\alpha\sigma\gamma \in X^k Z_0$. From Proposition 12 we conclude $\alpha\sigma\gamma \in Z^k$. ∎

Let M be an FSM and let $Z \subseteq X^\star$ be a set of input sequences. We indicate by $[Z]$ the partition induced by Z (see Definition 8) over the states of M, i.e, $s \approx_Z r$ if and only if $s, r \in w$, for some $w \in [Z]$.

The next result expresses properties of these partitions.

Lemma 14 *Let $\{Z_i\}_{i\geq 0}$ be a stratified family over the alphabet X of an FSM M. Then*

1. *$[Z_{i+1}]$ refines $[Z_i]$, for every $i \geq 0$; and*
2. *if $|[Z_k]| = |[Z_{k+1}]|$ for some $k \geq 0$, then $[Z_k] = [Z_{k+1}] = [Z_{k+2}]$.*

Proof We are going to show each item, in turn.

For item (1), assume it does not hold for some $i \geq 0$. Then we will have states s and r such that $s \approx_{Z_{i+1}} r$ and $s \not\approx_{Z_i} r$. From Lemmas 10(1) and 13(1) we deduce $s \not\approx_{Z_{i+1}} r$, a contradiction.

Now we verify item (2). From item (1), we know that $[Z_{k+1}]$ refines $[Z_k]$. Then $[Z_k] = [Z_{k+1}]$, otherwise we would have $|[Z_k]| < |[Z_{k+1}]|$. Again continuing by contradiction, assume that $[Z_{k+1}] \neq [Z_{k+2}]$. Since $[Z_{k+2}]$ refines $[Z_{k+1}]$, we will have states r and s such that $s \not\approx_{Z_{k+2}} r$ and $s \approx_{Z_{k+1}} r$. Hence, we obtain $\rho \in Z_{k+2}$, with $\rho = a\beta$ and $a \in X$, and such

that $s \not\approx_{a\beta} r$. We also conclude that $a\beta \notin Z_{k+1}$, otherwise we would have the contradiction $s \not\approx_{Z_{k+1}} r$. Therefore, from Definition 11(2), we deduce $a\beta \in XZ_{k+1}$, and so, $\beta \in Z_{k+1}$.

Let s_1, $r_1 \in S$ with $s_1 = \delta(a, s)$, $r_1 = \delta(a, r)$. If $s_1 \not\approx_{Z_{k+1}} r_1$ then $s_1 \not\approx_{Z_k} r_1$, because we already know that $[Z_k] = [Z_{k+1}]$. Hence, we would have $\gamma \in Z_k$ with $\widehat{\lambda}(\gamma, s_1) \neq \widehat{\lambda}(\gamma, r_1)$. From Definition 11(1) we have $XZ_k \subseteq Z_{k+1}$, and then $a\gamma \in Z_{k+1}$. But,

$$\widehat{\lambda}(a\gamma, s) = \lambda(a, s)\widehat{\lambda}(\gamma, s_1), \quad \text{and} \quad \widehat{\lambda}(a\gamma, r) = \lambda(a, r)\widehat{\lambda}(\gamma, r_1).$$

Then $\widehat{\lambda}(a\gamma, s) \neq \widehat{\lambda}(a\gamma, r)$, forcing the contradiction $s \not\approx_{Z_{k+1}} r$. We conclude that $s_1 \approx_{Z_{k+1}} r_1$.

Since $\beta \in Z_{k+1}$, we deduce $s_1 \approx_\beta r_1$. Again,

$$\widehat{\lambda}(a\beta, s) = \lambda(a, s)\widehat{\lambda}(\beta, s_1), \quad \text{and} \quad \widehat{\lambda}(a\beta, r) = \lambda(a, r)\widehat{\lambda}(\beta, r_1),$$

and, since we already have $\widehat{\lambda}(\rho, s) \neq \widehat{\lambda}(\rho, r)$, we conclude that $\lambda(a, s) \neq \lambda(a, r)$. From $a \in X$ and Lemma 13(2) we infer $\sigma \in X^\star$ with $a\sigma \in Z_{k+1}$. Hence, we have $s \not\approx_{a\sigma} r$, contradicting $s \approx_{Z_{k+1}} r$. ∎

The next result gives the equality of successive partitions.

Corollary 15 *Let $\{Z_i\}_{i\geq 0}$ be a stratified family over the input alphabet X of an FSM M. If $|[Z_k]| = |[Z_{k+1}]|$ for some $k \geq 0$, then $[Z_k] = [Z_{k+l}]$ for every $l \geq 0$.*

Proof When $l = 0$, the result is immediate. When $l = 1$ or $l = 2$, the result follows directly from Lemma 14(2). Assume the result holds for every j, $0 \leq j \leq l$, with $l \geq 2$. We want to show that the result holds for $l + 1$. From the induction, we have $[Z_k] = [Z_{k+l}]$ and $[Z_k] = [Z_{k+l-1}]$. Hence, $[Z_{k+l-1}] = [Z_{k+l}]$. Using Lemma 14(2), we obtain $[Z_{k+l-1}] = [Z_{k+l}] = [Z_{k+l+1}]$. Hence, $[Z_k] = [Z_{k+l+1}]$, as required. ∎

Now let M be an FSM with m states. Suppose we have a stratified family for X, $\{Z_i\}_{i\geq 0}$, in which Z_0 partitions the states of M in $n \leq m$ equivalence classes. We want to study the partitions over states of M induced by the Z_i sets, for $i \geq 0$. The next lemma establishes the basic result.

Lemma 16 *Let M be an FSM with index m. Let $\{Z_i\}_{i\geq 0}$ be a stratified family for X such that Z_0 partitions the states of M in at least $n \leq m$ equivalence classes. Then $|[Z_i]| \geq n+i$, for every i, $0 \leq i \leq m - n$.*

Proof When $i = 0$ we have $n + i = n$ and, from the hypothesis, $|[Z_0]| \geq n$, establishing the base. Assume the result for every j, $0 \leq j \leq i$, with $i < m - n$. We are going to show that the result holds for $i + 1$. If $|[Z_i]| \geq n + i + 1$ then $|[Z_{i+1}]| \geq n + i + 1$ (from Lemma 14(1)), and the induction is extended in this case.

Now, let $|[Z_i]| < n+i+1$. From the induction hypothesis we conclude that $|[Z_i]| = n+i$. Since $m \geq n + i + 1$ is the index of M, there exist nonequivalent states in M, r and s, with $r \approx_{Z_i} s$. Then, $s \not\approx_{X^k} r$, for some $k \geq 0$ (see Definition 8). From Lemma 13(2), we conclude

$s \not\approx_{Z_k} r$. If $k \leq i$, Lemma 13(1) would force $Z_k \subseteq Z_i$. Using Lemma 10(1) we would have $s \not\approx_{Z_i} r$, a contradiction. Hence, $k > i$.

If $|[Z_i]| = |[Z_{i+1}]|$ then, by Corollary 15, we would have $Z_i = Z_k$, forcing again the contradiction $s \not\approx_{Z_i} r$. Since $[Z_{i+1}]$ refines $[Z_i]$, we can not have $|[Z_{i+1}]| < |[Z_i]|$. We conclude that $|[Z_{i+1}]| > |[Z_i]|$. But, since $|[Z_i]| = n + i$, we deduce the result desired, that is, $|[Z_{i+1}]| \geq n + i + 1$. ∎

Using this result, it will be easy to confirm that some $Z \in \{Z_i\}_{i \geq 0}$ will distinguish every pair of nonequivalent states.

Corollary 17 *Let M be an FSM with index m. Let $\{Z_i\}_{i \geq 0}$ be a stratified family for X such that Z_0 partitions the states of M in at least $n \leq m$ equivalence classes. Then Z_{m-n} will distinguish every pair of nonequivalent states of M.*

Proof From Lemma 16, it follows that $|[Z_{m-n}]| \geq n + (m - n) = m$. Since $[Z_{m-n}]$ is the partition induced by Z_{m-n}, we conclude that Z_{m-n} partitions states of M in m classes. Since M has index m, we conclude that Z_{m-n} will distinguish every pair of nonequivalent states of M. ∎

5 An m-complete Test Suite

Let M and M' be two FSMs operating over the same alphabet X. Machine M would represent a specification and M' would represent a possible implementation. We wish a set $K \subseteq X^\star$ such that $s_0 \not\approx s'_0$ if and only if $s_0 \not\approx_K s'_0$. We say that K is an m-complete test suite, where m is an upper bound on the index of M'. Additionally, K must be kept as small as possible. In this scenario, if we want to test whether M and M' have distinct behaviors, it would be enough to apply the sequences in K to both machines M and M', and compare the corresponding outputs.

We are going to obtain the required set by combining a cover for M with a stratified family for M'. The next lemma establishes an auxiliary result.

Lemma 18 *Let M and M' be two FSMs operating over the same input alphabet, X. Assume that M' has index m and that P is a cover for M. Let $Z \subseteq X^\star$ be nonempty and such that Z partitions the states of M' in at least m equivalence classes. If $s_0 \approx_{PZ} s'_0$ and $s_0 \not\approx s'_0$, then there exist $\gamma \in X^\star$, $s \in S$, $s' \in S'$ such that $\widehat{\delta}(\gamma, s_0) = s$, $\widehat{\delta}'(\gamma, s'_0) = s'$ and $s \not\approx_Z s'$.*

Proof Since $s_0 \not\approx s'_0$ we obtain $\gamma \in X^\star$, $a \in X$, $s \in S$ e $s' \in S'$ such that $\widehat{\delta}(\gamma, s_0) = s$, $\widehat{\delta}'(\gamma, s'_0) = s'$ and $\lambda(a, s) \neq \lambda'(a, s')$. Since P is a cover for M, from edge $\delta(a, s) = r$ we obtain $\rho \in P$ e $\rho a \in P$ such that $\widehat{\delta}(\rho, s_0) = s$. We also know there exists $s'' \in S'$ such that $\widehat{\delta}'(\rho, s'_0) = s''$.

We are going to show that $s \approx_Z s''$. Otherwise we would have $\sigma \in Z$ with $\widehat{\lambda}(\sigma, s) \neq \widehat{\lambda}'(\sigma, s'')$. Hence, $\widehat{\lambda}(\rho\sigma, s_0) = \widehat{\lambda}(\rho, s_0)\widehat{\lambda}(\sigma, s)$ and $\widehat{\lambda}'(\rho\sigma, s'_0) = \widehat{\lambda}'(\rho, s'_0)\widehat{\lambda}'(\sigma, s'')$ and we would obtain $\widehat{\lambda}(\rho\sigma, s_0) \neq \widehat{\lambda}'(\rho\sigma, s'_0)$, contradicting $s_0 \approx_{PZ} s'_0$ since $\rho\sigma \in PZ$. We conclude $s \approx_Z s''$.

Now we are going to show that $s \not\approx_Z s'$. Otherwise we would have $s' \approx_Z s''$, since we already know that $s \approx_Z s''$. Since M' has index m and Z partitions the states of M' in m equivalence classes, we would obtain $s' \approx s''$. But then,

$$\begin{aligned} \widehat{\lambda}(\rho a, s_0) &= \widehat{\lambda}(\rho, s_0)\lambda(a, s) \\ \widehat{\lambda'}(\rho a, s'_0) &= \widehat{\lambda'}(\rho, s'_0)\lambda(a, s'') = \widehat{\lambda'}(\rho, s'_0)\lambda(a, s'). \end{aligned}$$

Since we already have $\lambda(a, s) \neq \lambda'(a, s')$, we deduce $\widehat{\lambda}(\rho a, s_0) \neq \widehat{\lambda'}(\rho a, s'_0)$, which gives us $s_0 \not\approx_P s'_0$ because $\rho a \in P$. Since Z is nonempty, Lemma 10(3) forces $s_0 \not\approx_{PZ} s'_0$, contradicting the hypothesis. We conclude that $s \not\approx_Z s'$.

Putting together, we have $\gamma \in X^\star$, with $\widehat{\delta}(\gamma, s_0) = s$, $\widehat{\delta'}(\gamma, s'_0) = s'$ e $s \not\approx_Z s'$. ∎

Now we are in a position to enunciate the result which will give us the capability of testing two machines for equivalence.

Theorem 19 *Let M and M' be two FSMs operating over the same input alphabet, X. Assume that M' has index m and that P is a cover for M. Let $Z \subseteq X^\star$ be nonempty and such that Z partitions states of M' in at least m equivalence classes. Then, $s_0 \approx s'_0$ if and only if $s_0 \approx_{PZ} s'_0$.*

Proof If $s_0 \approx s'_0$ then, trivially, $s_0 \approx_{PZ} s'_0$.

For the opposite direction, assume $s_0 \approx_{PZ} s'_0$. For the sake of contradiction, assume $s_0 \not\approx s'_0$. From Lemma 18, we obtain $\beta \in X^\star$, $s \in S$ and $s' \in S'$ with $\widehat{\delta}(\beta, s_0) = s$, $\widehat{\delta'}(\beta, s'_0) = s'$, and $s \not\approx_Z s'$. We can assume, without loss generality, that $|\beta|$ is minimal. If $\beta = \epsilon$, we would have $s = s_0$ and $s' = s'_0$, and then $s_0 \not\approx_Z s'_0$. But, since $\epsilon \in P$, this would force the contradiction $s_0 \not\approx_{PZ} s'_0$. We conclude that $\beta = \alpha a$, with $a \in X$. Let $r \in S$ and $r' \in S'$ with $\widehat{\delta}(\alpha, s_0) = r$, $\widehat{\delta'}(\alpha, s'_0) = r'$, $\delta(a, r) = s$ and $\delta'(a, r') = s'$. Using the minimality of $|\beta|$ we have $r \approx_Z r'$.

On the other hand, since P is a cover for M, from the edge $\delta(a, r) = s$ we obtain $\rho \in P$ and $\rho a \in P$ with $\widehat{\delta}(\rho, s_0) = r$. Let $r'' \in S'$ with $\widehat{\delta'}(\rho, s'_0) = r''$. If we had $r \not\approx_Z r''$, we would obtain $\gamma \in Z$ with $\widehat{\lambda}(\gamma, r) \neq \widehat{\lambda'}(\gamma, r'')$. But then

$$\widehat{\lambda}(\rho\gamma, s_0) = \widehat{\lambda}(\rho, s_0)\widehat{\lambda}(\gamma, r), \quad \text{and} \quad \widehat{\lambda'}(\rho\gamma, s'_0) = \widehat{\lambda'}(\rho, s'_0)\widehat{\lambda'}(\gamma, r'').$$

Hence, $\widehat{\lambda}(\rho\gamma, s_0) \neq \widehat{\lambda'}(\rho\gamma, s'_0)$, giving the contradiction $s_0 \not\approx_{\rho\gamma} s'_0$ with $\rho\gamma \in PZ$. We conclude that $r \approx_Z r''$.

Since we already have $r \approx_Z r'$, we obtain $r' \approx_Z r''$. Since Z partitions the states of M' in m classes and m is the index of M', we conclude that $r' \approx r''$. Now, from $s \not\approx_Z s'$, we obtain $\sigma \in Z$ with $\widehat{\lambda}(\sigma, s) \neq \widehat{\lambda'}(\sigma, s')$. But,

$$\begin{aligned} \widehat{\lambda}(\rho a\sigma, s_0) &= \widehat{\lambda}(\rho, s_0)\lambda(a, r)\widehat{\lambda}(\sigma, s) \quad \text{and} \\ \widehat{\lambda'}(\rho a\sigma, s'_0) &= \widehat{\lambda'}(\rho, s'_0)\widehat{\lambda'}(a\sigma, r'') = \widehat{\lambda'}(\rho, s'_0)\widehat{\lambda'}(a\sigma, r') = \widehat{\lambda'}(\rho, s'_0)\lambda'(a, r')\widehat{\lambda'}(\sigma, s'). \end{aligned}$$

Then, $\widehat{\lambda}(\rho a\sigma, s_0) \neq \widehat{\lambda'}(\rho a\sigma, s'_0)$. But $\rho a\sigma \in PZ$ and we would have $s_0 \not\approx_{PZ} s'_0$, contradicting the hypothesis. This concludes the proof. ∎

Combining the previous results, we have the following corollary, useful to determine whether two FSMs have distinguishing behaviors.

Corollary 20 *Let M and M' two FSMs operating over the same input alphabet, X. Assume M' has the index m. Assume also that P is a cover for M, that $R \subseteq X^\star$ is nonempty and that it partitions the states of M' in at least $n \leq m$ equivalence classes. Then, s_0 and s'_0 are equivalent if and only if s_0 and s'_0 are PZ-equivalent, where $Z = \bigcup_{i=0}^{m-n} X^i R$.*

Proof Let $Z_k = \bigcup_{i=0}^{k} X^i R$, $k \geq 0$. From Proposition 12 we have that such family $\{Z_k\}_{k\geq 0}$ is stratified. From Corollary 17 we conclude that Z distinguishes every pair of nonequivalent states of M'. Then the result follows directly from Theorem 19. ∎

6 Characterization Sets

From the previous corollary, it might appear that Z and machine M would be independent, since the only hypothesis involving M, in that corollary, is that P is a cover for M. However, such it is not the case. Before we expose the relationship between Z and M, we need another auxiliary result.

Lemma 21 *Let M and M' be two FSMs operating over the same input alphabet, X. Assume that all states of M are reachable and that $s_0 \approx s'_0$. Let $Z \subseteq X^\star$ be a set partitioning the states of M' in m equivalence classes, where m is the index of M'. Then Z distinguishes every pair of nonequivalent states of M.*

Proof Let s_1, $s_2 \in S$ with $s_1 \not\approx s_2$ and assume $s_1 \approx_Z s_2$. Since all states of M are reachable, we have ρ_1 and $\rho_2 \in X^\star$ such that $\widehat{\delta}(\rho_i, s_0) = s_i$, with $i = 1, 2$. In M' we would have some $s'_1, s'_2 \in S'$ and with $\widehat{\delta'}(\rho_i, s'_0) = s'_i$, where $i = 1, 2$.

Now let $\beta \in Z$. We have,

$$\widehat{\lambda}(\rho_2\beta, s_0) = \widehat{\lambda}(\rho_2, s_0)\widehat{\lambda}(\beta, s_2), \quad \text{and} \quad \widehat{\lambda'}(\rho_2\beta, s'_0) = \widehat{\lambda'}(\rho_2, s'_0)\widehat{\lambda'}(\beta, s'_2)$$

and, since $s_0 \approx s'_0$, we obtain $\widehat{\lambda}(\beta, s_2) = \widehat{\lambda'}(\beta, s'_2)$ and $\widehat{\lambda}(\rho_2, s_0) = \widehat{\lambda'}(\rho_2, s'_0)$. Since β is arbitrary, we conclude that $s_2 \approx_Z s'_2$.

Similarly,

$$\widehat{\lambda}(\rho_1\beta, s_0) = \widehat{\lambda}(\rho_1, s_0)\widehat{\lambda}(\beta, s_1), \quad \text{and} \quad \widehat{\lambda'}(\rho_1\beta, s'_0) = \widehat{\lambda'}(\rho_1, s'_0)\widehat{\lambda'}(\beta, s'_1),$$

and we conclude that $s_1 \approx_Z s'_1$, together with $\widehat{\lambda}(\rho_1, s_0) = \widehat{\lambda'}(\rho_1, s'_0)$.

Putting it together, and knowing that $s_1 \approx_Z s_2$, we obtain $s_1 \approx_Z s'_2$ and also $s_2 \approx_Z s'_1$. Hence, $s'_1 \approx_Z s'_2$. But s'_1 and s'_2 are states of M' and so the hypothesis over Z gives $s'_1 \approx s'_2$.

On the other hand, since $s_1 \not\approx s_2$, we obtain $\sigma \in X^\star$ such that $\widehat{\lambda}(\sigma, s_1) \neq \widehat{\lambda}(\sigma, s_2)$. Now,

$$\widehat{\lambda}(\rho_1\sigma, s_0) = \widehat{\lambda}(\rho_1, s_0)\widehat{\lambda}(\sigma, s_1), \quad \text{and} \quad \widehat{\lambda'}(\rho_1\sigma, s'_0) = \widehat{\lambda'}(\rho_1, s'_0)\widehat{\lambda'}(\sigma, s'_1).$$

Hence, from $\widehat{\lambda}(\rho_1\sigma, s_0) = \widehat{\lambda'}(\rho_1\sigma, s'_0)$ and from $\widehat{\lambda}(\rho_1, s_0) = \widehat{\lambda'}(\rho_1, s'_0)$, we deduce $\widehat{\lambda}(\sigma, s_1) = \widehat{\lambda'}(\sigma, s'_1)$.

Similarly,

$$\widehat{\lambda}(\rho_2\sigma, s_0) = \widehat{\lambda}(\rho_2, s_0)\widehat{\lambda}(\sigma, s_2), \quad \text{and} \quad \widehat{\lambda'}(\rho_2\sigma, s'_0) = \widehat{\lambda'}(\rho_2, s'_0)\widehat{\lambda'}(\sigma, s'_2),$$

and then $\widehat{\lambda}(\sigma, s_2) = \widehat{\lambda'}(\sigma, s'_2)$. However, since we already know that $s'_1 \approx s'_2$ and this leads to the contradiction $\widehat{\lambda}(\sigma, s_1) = \widehat{\lambda}(\sigma, s_2)$. This shows that the initial hypothesis was false. Hence, whenever $s_1 \not\approx s_2$ holds we must also have $s_1 \not\approx_Z s_2$, establishing the result. ∎

A set in these conditions is called a characterization set of M.

Definition 22 *Let M be an FSM and W a set of input sequences. W is a characterization set for M if W distinguishes any pair of nonequivalent states of M.*

The required relation between M and Z says that Z is a characterization set of M, under certain hypothesis.

Theorem 23 *Let M and M' be two FSMs operating over the same input alphabet, X. Assume that M' has index m and that P is a cover for M. Assume also that $W \subseteq X^\star$ is nonempty and partitions the states of M' in at least $n \leq m$ equivalence classes. If $s_0 \approx_{PZ} s'_0$ then $Z = \bigcup_{i=0}^{m-n} X^i W$ is a characterization set for M.*

Proof From Proposition 12 and from Corollary 17 we conclude that Z distinguishes every pair of nonequivalent states of M'. Since P is cover for M, we conclude that every state of M is reachable. From $s_0 \approx_{PZ} s'_0$, together with Corollary 20, we deduce $s_0 \approx s'_0$. Now we can use Lemma 21 and obtain that Z distinguishes every pair of nonequivalent states of M. From Definition 22, Z is a characterization set for M. ∎

It is also easy to see that the reverse does not hold. For that, let M and M' be two FSMs. It is clear that $W = X^\star$ partitions the states of M and of M' in the maximum number of equivalence classes. In this case, we will have $Z = W = X^\star$ and, obviously, Z is a characterization set for M and M'. But it is not the case that we will always have $s_0 \approx s'_0$, as it is easy to construct a counter-example.

Next result shows that, under relaxed conditions, when two FSMs are equivalents both must have the same index.

Theorem 24 *Let M and M' be two FSMs operating over the same input alphabet, X. Let n and n' be the index of M and of M', respectively. Assume that all states from both FSMs are reachable. If $s_0 \approx s'_0$ then $n = n'$.*

Proof For the sake of contradiction, and without loss generality, we will assume $n < n'$.

Let $s'_i \in S'$, $1 \leq i \leq n'$, be states from each one of n' equivalence classes induced by $\approx$ in S'. Since all states of M' are reachable, we obtain $\rho_i \in X^\star$ with $\widehat{\delta'}(\rho_i, s'_0) = s'_i$, $1 \leq i \leq n'$. In M, we will have some $s_i \in S$ such that $\widehat{\delta}(\rho_i, s_0) = s_i$, $1 \leq i \leq n'$. Since $n < n'$, without loss generality, we can say that $s_1 \approx s_2$.

Take any $z \in X^\star$. We have

$$\widehat{\lambda}(\rho_1 z, s_0) = \widehat{\lambda}(\rho_1, s_0)\widehat{\lambda}(z, s_1), \quad \text{and} \quad \widehat{\lambda'}(\rho_1 z, s'_0) = \widehat{\lambda'}(\rho_1, s'_0)\widehat{\lambda'}(z, s'_1).$$

Since $s_0 \approx s'_0$, it follows that $\widehat{\lambda}(z, s_1) = \widehat{\lambda}'(z, s'_1)$. Similarly, $\widehat{\lambda}(z, s_2) = \widehat{\lambda}'(z, s'_2)$.

But since $s_1 \approx s_2$, we obtain $\widehat{\lambda}(z, s_1) = \widehat{\lambda}(z, s_2)$. Therefore, $\widehat{\lambda}'(z, s'_1) = \widehat{\lambda}'(z, s'_2)$. Since $z \in X^\star$ is arbitrary, we conclude that $s'_1 \approx s'_2$, a contradiction given that s'_1 and s'_2 are in distinct classes in M'.

Hence, we must have $n \geq n'$. Similarly, $n' \geq n$, and then $n = n'$. ∎

The same result shows that when the $\approx$ relation induces a different number of equivalence classes in two FSMs, then these machines can not be equivalent to each other (under the weak hypothesis of Theorem 24). On the other hand, it is simple to obtain two nonequivalent FSMs, in a such way that the $\approx$ relation induces the same number of equivalence classes in both machines.

7 The W-method as a Particular Case of the Generalization

From Theorem 23, we can show that W is a characterization set of M if we have n as the index of M and the behaviors match.

Corollary 25 *Let M and M' be two FSMs operating over the same input alphabet, X, and that all states in M' are reachable. Assume further that M' has index m, that P is a cover for M and that M has index n. Assume also that $W \subseteq X^\star$ is nonempty and partitions the states of M' in at least $n \leq m$ equivalence classes. If $s_0 \approx_{PZ} s'_0$, where $Z = \bigcup_{i=0}^{m-n} X^i W$, then $n = m$, $Z = W$ and W is a characterization set for M.*

Proof Since $s_0 \approx_{PZ} s'_0$, together with Corollary 20, we conclude that $s_0 \approx s'_0$. Next, we infer that $n = m$, from Theorem 24. Hence, $Z = W$. Therefore, by Theorem 23, W is a characterization set for M. ∎

When using our more general method (see Corollary 20), all we need is an upper bound on the number of classes in M' induced by W.

On the other hand, when W is a characterization set of M, we guarantee the partitioning of M' in a number of classes at least equal to the index of M, if the machines are to be PZ-equivalent.

Lemma 26 *Let M and M' be two FSMs operating over the same input alphabet, X. Assume that M' has index m, that M has index n and that P is a cover for M, with $n \leq m$. Assume also that $W \subseteq X^\star$ is a characterization set for M and that $s_0 \approx_{PZ} s'_0$, where $Z = \bigcup_{i=0}^{m-n} X^i W$. Then W partitions M' in at least n equivalence classes.*

Proof We know that M has n equivalence classes: Let $C_1, \ldots, C_n$ be these classes. Let $s_i \in C_i$ and $s_j \in C_j$, where $1 \leq i < j \leq n$. Then since W is a characterization set for M, we have $s_i \not\approx_W s_j$. Since P is cover of M, we have $\widehat{\delta}(\rho, s_0) = s_i$, for some $\rho \in P$. We also know that $\widehat{\delta}'(\rho, s'_0) = s'_i$, for some s'_i of M'. Since $s_0 \approx_{PZ} s'_0$, we get $s_i \approx_Z s'_i$. Since $W \subseteq Z$, then $s_i \approx_W s'_i$. In the same way, we have s'_j of M' with $s_j \approx_W s'_j$. Then we obtain $s'_i \not\approx_W s'_j$, otherwise $s_i \approx_W s_j$. We conclude that W partitions M' in at least $n \leq m$ equivalence classes. ∎

Now we can use Lemma 26 to show the Theorem 27, since W partitions M' in at least the same number of classes in M.

Theorem 27 *Let M and M' be two FSMs operating over the same input alphabet, X. Assume that M' has index m, that P is a cover for M and that M has index n, with $n \leq m$. Assume also that $W \subseteq X^\star$ is a characterization set for M and that $s_0 \approx_{PZ} s'_0$, where $Z = \bigcup_{i=0}^{m-n} X^i W$. Then $s_0 \approx s'_0$.*

Proof Assume $s_0 \approx_{PZ} s'_0$. Use Lemma 26 to show that W partitions M' in at least n classes. Now use Corollary 17 to show that Z partitions M' in m classes. Finally, use Theorem 19. ∎

The next result is the main postulate of the basic W-method, as given in [Cho78].

Theorem 28 *Let M and M' be two FSMs operating over the same input alphabet, X. Assume that M' has index m, that P is a cover for M and that M has index n, with $n \leq m$. Assume also that $W \subseteq X^\star$ is a characterization set for M. Then $s_0 \approx s'_0$ if and only if $s_0 \approx_{PZ} s'_0$, where $Z = \bigcup_{i=0}^{m-n} X^i W$.*

Proof If $s_0 \approx s'_0$, then $s_0 \approx_{PZ} s'_0$, trivially. For the other direction, use Theorem 27. ∎

In general (Corollary 20), W does not need to be a characterization set for M. In that case we need only the guarantee that M' will be partitioned in at least some n equivalence classes, with $n \leq m$, for the method to work, where m is the index of M'. There need be no relationship between W and M. On the other hand, when using the W-method directly, we need to obtain a characterization set W for M, and we need to know also the index of M, and secure the relationship $n \leq m$. When W is not a characterization set for M, the method may fail, as shown by the following example.

The alphabet of M and M' is $X = \{a, b, c\}$. See Figures 1 and 2.

It is easy to see that M has index $n = 3$, because $s_1 \approx s_3$. The index of M' is $m = 3$ since $s'_1 \approx s'_3 \approx s'_4$. Hence $m = n$, and we would be left with $Z = W$.

Now take $W = \{\epsilon\}$.

A cover set is given by $P = \{\epsilon, aa, ab, ac, ba, bb, bc, ca, cb, cc\}$. Then, $PZ = PW = P$.

It is easy to see that M and M' are equivalent. But $s_0 \approx s'_0$ is not true. To see that, take $\alpha = bbb$. We have $\widehat{\lambda}(\alpha, s_0) = 100$ and $\widehat{\lambda}'(\alpha, s'_0) = 101$.

Note how W induces only one equivalence class in M'. Clearly, W is not a characterization set for M.

In general, it would be important to devise a mechanism by which we could obtain the number of classes induced by W in M'. First, because in this case we might avoid calculating a characterization set for M, when using our more general method. Secondly, we could potentially diminish the size of the sequences in Z, when W partitions M' in k classes, with $k > n$, given that $Z = \bigcup_{i=0}^{m-n} X^i W$.

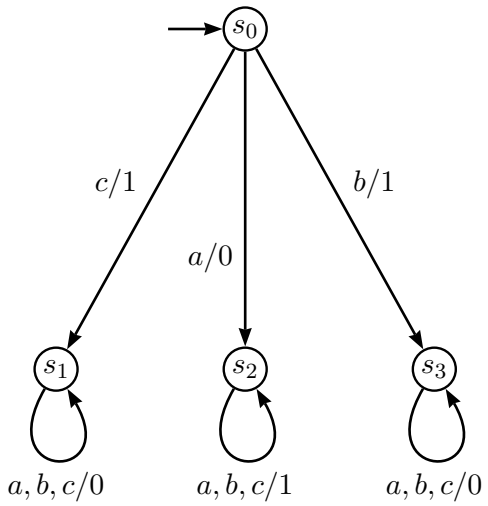

Figure 1: Specification M for the counter-example.

8 The Generalized Test Case Generation Method

Algorithm 1 presents the generalized model-based test generation method. The input parameters are:

- M represents a system specification;
- M' is an implementation candidate for the specification M;
- R is any set of input sequences;
- n is a lower bound on the number of classes induced by R in M'; and
- m is an upper bound on the index of M'.

The method requires knowledge of a lower bound on the number n of equivalence classes induced by R in the implementation machine M', as well as an upper bound on the index m of M'. In an extreme case, once can set $n = 1$ and $m = |S'|$, that is, set m to the number of states in M'. Note that the implementation M' is given as a black box. So, we do not have access to its internal structure, and the parameters n and m must be estimated. As for the specification M, R may partition it in any number k of classes. Of course, if M and M' turn out to be equivalent, then they will have the same index and Z will, in fact, be a characterization set for both M and M'.

If the condition $n \leq m$ is secured and it turns out that M and M' are not equivalent, the algorithm produces a particular input sequence σ that is a witness to this fact, that is, M and M' display distinct behaviors over σ.

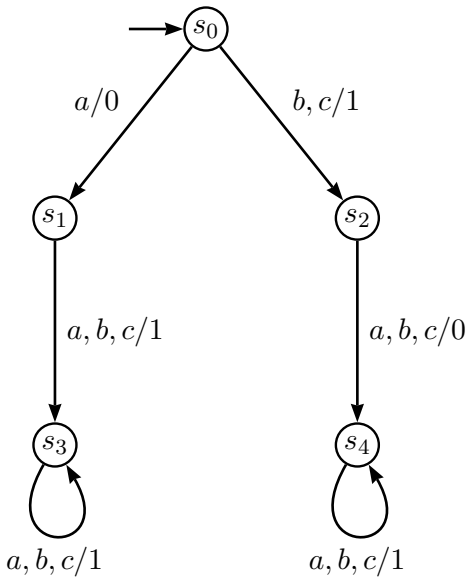

Figure 2: Implementation candidate M' for the counter-example.

Algorithm 2 presents the basic W-method. Note that, in this case, we have to compute the index of M as well as a characterization set for M. Therefore, some extra effort must be applied in order to obtain n and W.

In our proposal, using practical information about M one can obtain a good candidate for R. Some distinguishing sequences can be inserted into R, based on the number of symbols in the input alphabet and on the number of states and transitions in M. Then, it is easy to obtain the set Z using the notion of stratification.

Clearly, also in the basic W-method as well, after obtaining the concatenation PZ, we can use this product to verify conformance between the specification and several proposed implementations.

9 A Simple Example

Now we show the application of generalized algorithm in a simple example. Let a specification M given as in Figure 3. We have M with $k = 4$ states, the input alphabet is $X = \{a, b\}$, the output alphabet is $Y = \{0, 1\}$, and the transition function is given as depicted the figure. We can see that some transitions over the input a produce either output 0 or output 1. Hence, there are at least two states in distinct classes. Now, if we use the sequences aa and ba, there is a good chance that such sequences can distinguish other states. Therefore, we take $R = \{aa, ba\}$ and assume that R partitions M', an implementation candidate, in at least $n = 3$ equivalence classes. Take $m = 5$ as a maximum on the number of states in M', and we have all conditions for Algorithm 1 secured. Note that R is not a characterization set for M because we have states s_0 and s_1 in the same equivalence

Algorithm 1: Generalized test generation algorithm.

```
Input: M, M', R, m, n
begin
    Obtain a cover set P for M;
    if n ≤ m then
        Compute Z = ⋃_{i=0}^{m−n} X^i R;
        Compute PZ;
    else
        mesg: M and M' are not equivalent;
        return ;
    end
    foreach σ ∈ PZ do
        Apply σ to M and to M';
        Obtain y = λ̂(σ, s_0) and y' = λ̂'(σ, s'_0);
        if y ≠ y' then
            mesg: M and M' are not equivalent;
            mesg: σ is an input witness;
            return ;
        end
    end
    mesg: M and M' are equivalents;
    return ;
end
```

class induced by R. Next we calculate a cover set P for M. In the example, we obtained the cover $P = \{\epsilon, a, b, aa, ab, ba, bb, aab, aaa\}$, using the labeled tree construction (see Section 3). Now we compute $Z = \bigcup_{i=0}^{m-n} X^i R$, where $m = 5$, $n = 3$ and $R = \{aa, ba\}$. We obtain $Z = \{aaaa, abaa, baaa, bbaa, aaba, abba, baba, bbba, aaa, aba, baa, bba, aa, ba\}$. Then, the concatenation PZ will count 56 sequences, not considering prefixes, and totalling 312 input symbols together with resets.

10 Concluding Remarks

Deriving test cases using formal models give us a rigorous validation of systems' behaviors. FSM is a well-established model that have been intensively investigated for this purpose. The W-method is a well known technique to compute test sequences using FSMs as its basic formal models. In this paper we extended and generalized such method, by relaxing the initial conditions of having computed a characterization set for the specification, as well as its index. In our method, we do not need a characterization set, nor the index of the specification.

In summary, our contribution is threefold. First, we generalized the basic W-method, avoiding the computation of characteristic sets and indexes. Secondly, we demonstrated in a clear way how the basic W-method follows from our generalized method. And finally, we

Algorithm 2: The basic W-method.

Input: M, M', m
begin
  Obtain a cover set P;
  Obtain a characterization set W for M;
  Obtain the index n of M;
  if $n \leq m$ **then**
    Compute $Z = \bigcup_{i=0}^{m-n} X^i W$;
    Compute PZ;
  else
    mesg: M and M' are not equivalent;
    return ;
  end
  foreach $\sigma \in PZ$ **do**
    Apply σ to M and to M';
    Obtain $y = \widehat{\lambda}(\sigma, s_0)$ and $y' = \widehat{\lambda}'(\sigma, s'_0)$;
    if $y \neq y'$ **then**
      mesg: M and M' are not equivalent;
      mesg: σ is an input witness;
      return ;
    end
  end
  mesg: M and M' are equivalent;
  return ;
end

presented detailed proofs of correctness both of our algorithm, as well as for main tenets of the basic W-method, the latter being absent in the original work that introduced the basic W-method.

As future steps in this research we plan to integrate results presented in this paper with the extension of EFSM [PBG04] proposed in [BMdSSaM08], where time constraints are an important feature to be considered in the models. Given that, we intend to use such ideas to generalize other methods, e.g. W_p and HSI.

References

[BMdSSaM08] Adilson Luiz Bonifácio, Arnaldo Vieira Moura, Adenilso da Silva Simão, and José Carlos Maldonado. Towards deriving test sequences by model checking. *Electron. Notes Theor. Comput. Sci.*, 195:21–40, 2008.

[Cho78] T. S. Chow. Testing software design modeled by finite-state machines. *IEEE Transactions on Software Engineering*, 4(3):178–187, 1978.

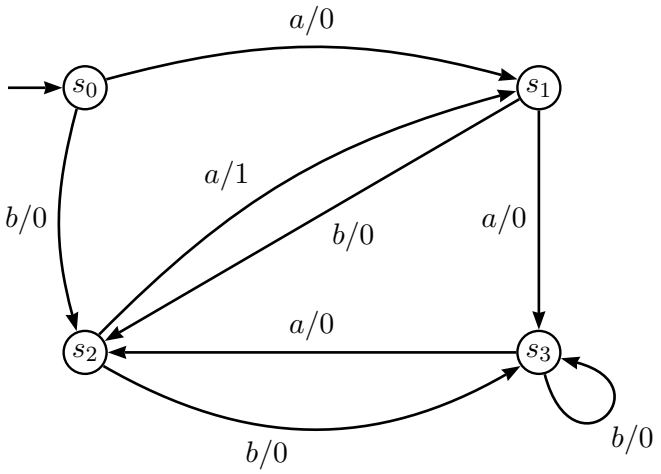

Figure 3: Machine specification M.

[CR05] Steven J. Cunning and Jerzy W. Rozenblit. Automating test generation for discrete event oriented embedded system s. *J. Intell. Robotics Syst.*, 41(2-3):87–112, 2005.

[DEFY05] Rita Dorofeeva, Khaled El-Fakih, and Nina Yevtushenko. An improved conformance testing method. In *FORTE*, pages 204–218, 2005.

[FBK+91] S. Fujiwara, G. V. Bochmann, F. Khendek, M. Amalou, and A. Ghedamsi. Test selection based on finite state models. *IEEE Transactions on Software Engineering*, 17(6):591–603, June 1991.

[Gar05] A. Gargantini. Conformance testing. In M. Broy, B. Jonsson, J.-P. Katoen, M. Leucker, and A. Pretschner, editors, *Model-Based Testing of Reactive Systems: Advanced Lectures*, volume 3472 of *Lecture Notes in Computer Science*, pages 87–111. Springer-Verlag, 2005.

[Gil62] A. Gill. *Introduction to the theory of finite-state machines.* McGraw-Hill, New York, 1962.

[Gon70] G. Gonenc. A method for the design of fault detection experiments. *IEEE Trans. Comput.*, 19(6):551–558, 1970.

[Hen64] F. C. Hennie. Fault detecting experiments for sequential circuits. In *FOCS*, pages 95–110, 1964.

[Hie06] R. M. Hierons. Separating sequence overlap for automated test sequence generation. *Automated Software Engg.*, 13(2):283–301, 2006.

[Kri05] M. Krichen. State identification. In M. Broy, B. Jonsson, J.-P. Katoen, M. Leucker, and A. Pretschner, editors, *Model-Based Testing of Reactive*

Systems: Advanced Lectures, volume 3472 of *Lecture Notes in Computer Science*, pages 87–111. Springer-Verlag, 2005.

[LvBP94] Gang Luo, G. von Bochmann, and A. Petrenko. Test selection based on communicating nondeterministic finite-state ma chines using a generalized wp-method. *IEEE Trans. Softw. Eng.*, 20(2):149–162, 1994.

[Mye79] G. J. Myers. *The Art of Software Testing*. Wiley, 1979.

[NS01] Brian Nielsen and Arne Skou. Test generation for time critical systems: Tool and case study. *ecrts*, 00:0155, 2001.

[PBG04] Alexandre Petrenko, Sergiy Boroday, and Roland Groz. Confirming configurations in efsm testing. *IEEE Trans. Softw. Eng.*, 30(1):29–42, 2004.

[PvB95] Alexandre Petrenko and Gregor v. Bochmann. Selecting test sequences for partially-specified nondeterministic finite state machines. In Gang Luo, editor, *IWPTS '94: 7th IFIP WG 6.1 international workshop on Protocol test systems*, pages 95–110, London, UK, UK, 1995. Chapman & Hall, Ltd.

[RU95] Ali Rezaki and Hasan Ural. Construction of checking sequences based on characterization sets. *Computer Communications*, 18(12):911–920, 1995.

[SkL89] Deepinder P. Sidhu and Ting kau Leung. Formal methods for protocol testing: A detailed study. *IEEE Trans. Softw. Eng.*, 15(4):413–426, 1989.

[SL88] D. Sidhu and T. Leung. Experience with test generation for real protocols. In *SIGCOMM '88: Symposium proceedings on Communications architectures and protocols*, pages 257–261, New York, NY, USA, 1988. ACM.

[Tre96] Jan Tretmans. Test generation with inputs, outputs, and quiescence. In Tiziana Margaria and Bernhard Steffen, editors, *Tools and Algorithms for Construction and Analysis of Systems, Second International Workshop, TACAS '96, Passau, Germany, March 27-29, 1996, Proceedings*, volume 1055 of *Lecture Notes in Computer Science*, pages 127–146. Springer, 1996.

[Tre99] Jan Tretmans. Testing concurrent systems: A formal approach. In J.C.M Baeten and S. Mauw, editors, *CONCUR '99: Proceedings of the 10th International Conference on Co ncurrency Theory*, volume 1664 of *Lecture Notes in Computer Science*, pages 46–65, London, UK, 1999. Springer-Verlag.