Joint NCC & IIS Bull., Comp. Science, 18 (2002), 69–80
© 2002 NCC Publisher

Novel modification of the W-method

M. Dorofeeva, I. Koufareva

Testing strategies based on finite states machines (FSMs) are widely used for protocol test derivation. Most FSM-based methods of test generation are based on the well-known W-method. In this paper, we determine the necessary conditions for a test suite to be complete under an assumption that only the number of states of an implementation under test is known. Based on the above conditions, we propose a novel modification of the W-method for the complete test suite derivation. We also present the results of computer experiments that clearly show that the total length of a test suite returned by the proposed modification is close to the minimal.

1. Introduction

In the protocol conformance testing, there exist a number of test derivation strategies based on finite state machines (FSMs). The FSMs are used to represent both the specification and the implementation under test (IUT). Usually a test suite has to determine whether the IUT is equivalent to its specification, i.e., if both machines have the same behavior. Some methods are based on an explicit enumeration [1] of all possible implementations, while another methods do not use an explicit enumeration of faulty machines. The former methods usually return a test suite of minimal total length, but they are out of practical use when the fault domain is large enough.

Most methods that do not use an explicit enumeration of all faulty machines are based on the well-known W-method [2, 3]. There exist some modifications of the W-method, in particular, *Wp*- and *HSI*-methods [4, 5] that return a shorter test suite than the original method. It can be shown that identification facilities of the states of the specification FSM is the only thing that can be manipulated in order to shorten the obtained test suite. However, the paper [6] has an example illustrating that the arbitrary state identifiers do not always preserve completeness of a test suite. Moreover, in all modifications the authors use the fixed state identifiers throughout the test suite construction process.

In this paper, we show that, in the case when only the number of states of an IUT is known, a complete test suite always has an appropriate identification sequence for each transition of the implementation. The sequence has a prefix that takes the implementation to an appropriate state; then an appropriate input is submitted. Finally, the suffix of the sequence distinguishes the expected final state of the transition from any other state of the specification. Based on the above condition, we propose to use different identification sequences for the same state. Identification sequences in our modification depend on a sequence that takes the specification FSM to a given state. The result essentially depends on a strategy for selecting the distinguishing sequences. The computer experiments we performed clearly show that the new modification returns a test suite shorter than that returned by the W- and *HSI*-methods. Moreover, the length of the obtained test suite is close to the lower bound [7].

The rest of the paper is structured as follows. Preliminaries are in Section 2, while Section 3 briefly describes the enumeration, W- and *HSI*-methods for generating test cases from a given FSM specification. The necessary conditions for a test suite to be complete when only the upper bound on the number of states of an implementation under test is known are presented in Section 4. A novel modification of the W-method is described in Section 5. Section 6 concludes the paper.

2. Finite state machines

Protocols can be formally considered as systems mapping the sequences of input symbols into the sequences of output symbols. Any finite set of symbols is called an *alphabet*. For any alphabet X, X^m denotes the set of all words of length m over alphabet X, while X^* is the set of all finite words including the empty word ε. Any two words u and v from X^* can be *concatenated* and produce a new word $uv \in X^*$. Given a subset $U \subseteq X^*$, the set $Pref(U)$ is the set of all prefixes of all words of U. For any two sets $U, V \subseteq X^*$, the set $UV \subseteq X^*$ is called the *concatenation* of U and V and is defined as $UV = \{uv | u \in U, v \in V\}$.

A *finite state machine* (FSM) A, called a *machine* throughout this paper, is a *deterministic complete initialized finite state machine*, i.e., a 6-tuple (S, X, Y, δ, λ, s_0), where S is a finite nonempty set of states with the initial state s_0, X and Y are input and output alphabets, $\delta : S \times X \to S$ is a *next state* function, and $\lambda : S \times X \to Y$ is an *output* function. In the usual way, the functions δ and λ are extended to words in X^*.

The FSM A is called *connected* if for each state $s \in S$ there exists an input sequence $\alpha_s \in X^*$ that takes the FSM A from the initial state to the state s; α_s is called a *transfer* sequence for the state s. The set of input sequences that has an empty sequence and a transfer sequence for each state is called a *state cover set* of the FSM A. We assume that there exists a special reliable input r called *reset input* that takes the machine from any state to the initial state.

A state s of an FSM A and a state t of an FSM $B = (T, X, Y, \Delta, \Lambda, t_0)$ are said to be *equivalent*, written $s \cong t$, if for all $\alpha \in X^*$ it holds $\lambda(s, \alpha) = \Lambda(t, \alpha)$. Otherwise, we say that the states s and t are *distinguishable* and an input sequence β such that $\lambda(s, \beta) \neq \Lambda(t, \beta)$ is called a *distinguishing* sequence for the states s and t. An FSM A is said to be *reduced* or *minimal* if its states are pair-wise distinguishable. The machines A and $B = (T, X, Y, \psi, \varphi, t_0)$ are *equivalent*, written $A \cong B$, if their initial states are equivalent; otherwise, the machines are called *distinguishable*. The sequence α such that $\lambda(s_0, \alpha) \neq \varphi(t_0, \alpha)$ is said to *distinguish* the FSM A from the FSM B.

A *fault domain* is a set of all possible faulty implementations. Based on the assumption that only the number of states of an implementation under test is known, we consider the fault domain consisting of all machines with at most m states over the input alphabet X. The set is denoted as $J_m(X)$. A set $TS \subset X^*$ is an *m-complete test suite* for an FSM A [8] if for each machine $B \in J_m(X)$ that is distinguishable from A, there exists a sequence in TS distinguishing the machine A from B.

In the next section, we sketch three methods for an m-complete test suite derivation, analyze them and show how these methods can be enhanced.

3. A sketch of the test derivation methods

1. Explicit enumeration of faulty machines. A straightforward approach to derivation of an m-complete test suite from a specification FSM A is an explicit enumeration of all machines of $J_m(X)$ over the output alphabet Y. For each FSM, we determine the shortest sequence distinguishing the machine from the specification FSM. The set of all such sequences is an m-complete test suite for A. A test so derived is called an *enumeration test*. The test is known to have little redundancy and its length is close to minimal; however, the latter essentially depends on the order of the FSM enumeration. It is known that the length of the shortest sequence distinguishing two machines does not exceed $m + n - 1$. By this reason, it is naturally to expect that the test length increases with n when m is fixed. The main disadvantage of the enumeration method is its time consumption due to a huge number of machines in the fault domain. Other well-known methods return an m-complete test suite without explicit enumeration of faulty machines.

2. ***W*- and *HSI*-methods.** The above testing methods use input sequences that can distinguish the states of a specification FSM in order to check the equivalence between transitions of the specification FSM and corresponding transitions of an implementation under test. Given a reduced FSM $A = (S, X, Y, \delta, \lambda, s_0)$ and a state s_i of the FSM A, a set W_i of input sequences is called a *state identifier* of the state s_i if for any other state s_j there exists $\alpha \in W_i$ such that $\lambda(s_i, \alpha) \neq \lambda(s_j, \alpha)$. We now define a family of *harmonized identifiers* [5] or a *separating family.* A family of harmonized state identifiers is a collection of state identifiers $W_i, s_i \in S$, which satisfy the condition that for any two states s_i, and s_j, $i \neq j$, there exist $\beta \in W_i$ and $\gamma \in W_j$ with a common prefix α such that $\lambda(s_i, \alpha) \neq \lambda(s_j, \alpha)$. A *characterization set* of the FSM A is a union of identifiers of each state. A characterization set and a family of harmonized state identifiers always exist for a reduced machine.

Given a specification reduced FSM $A = (S, X, Y, \delta, \lambda, s_0)$, $|S| = n$, and an implementation FSM $B = (T, X, Y, \Delta, \Lambda, t_0)$ such that $|T| = m$, let W be a characterization set of A and $F = \{W_1, \dots, W_n\}$ be a family of harmonized state identifiers of A.

All the methods have three phases. In the first phase, they establish a mapping $h : T \to S$ using the set $VPref(X^{m-n})$, where V is a state cover set of the specification machine, and $Pref(X^{m-n})$ is the set of all sequences of length at most $m - n$ over the alphabet X. After that, all inputs $x \in X$ are submitted to verify transitions from the reached states of the implementation. The third phase is meant to check if any state t of the implementation is equivalent to the corresponding state $s = h(t)$ of the specification, and if the same holds for the next states of all transitions from t. For this purpose, for each sequence $\alpha_j \in VPref(X^{m-n+1})$ a test suite has a subset $r\alpha_j W_j$ (*HSI*-method) or $r\alpha_j W$ (*W*-method), where r is the reset input. If an FSM B passes the test sequences of all testing phases, then it is equivalent to the specification FSM A, i.e., it is a *conforming* implementation.

The length of a test suite derived by the W-method does not exceed $n^2 \cdot m \cdot |X|^{m-n+1}$ [2]. Thus the total length of the test suite exponentially depends on $m - n$ and, at fixed m, the less is n, the greater is the length of the test suite. The latter contradicts the above assumption that the length of an enumeration test increases with n at fixed m. Therefore, it is interesting to study if this fact is a drawback of the W-method or it is natural. It is clear that the length of the test derived by the *HSI*-method does not exceed the length of the test derived by W-method. However, by construction, a test suite includes the set $VPref(X^{m-n+1})$, i.e., the test length also exponentially depends on $(m - n)$ and decreases with increasing n at fixed m.

Below we present the results of computer experiments [9]. Figure 1 represents the average length of a 6-complete test suite derived by the above methods. One can easily see that the total length of m-complete test suite exponentially depends on the difference between $(m - n)$ and deteriorates with increasing n at fixed m, no matter what method is used to derive the test. The latter contradicts a natural supposition that the more complex is the specification FSM the more complex is a test if the implementation is equivalent to the specification. One of possible explanations of the obtained results is that the test cases become more powerful when n is close to m, i.e., each test case detects more faulty machines of $J_m(X)$.

Unfortunately, we could not precisely estimate the length of an enumeration m-complete test suite for greater values of m, since the number of machines of the set $J_m(X)$ is exponential.

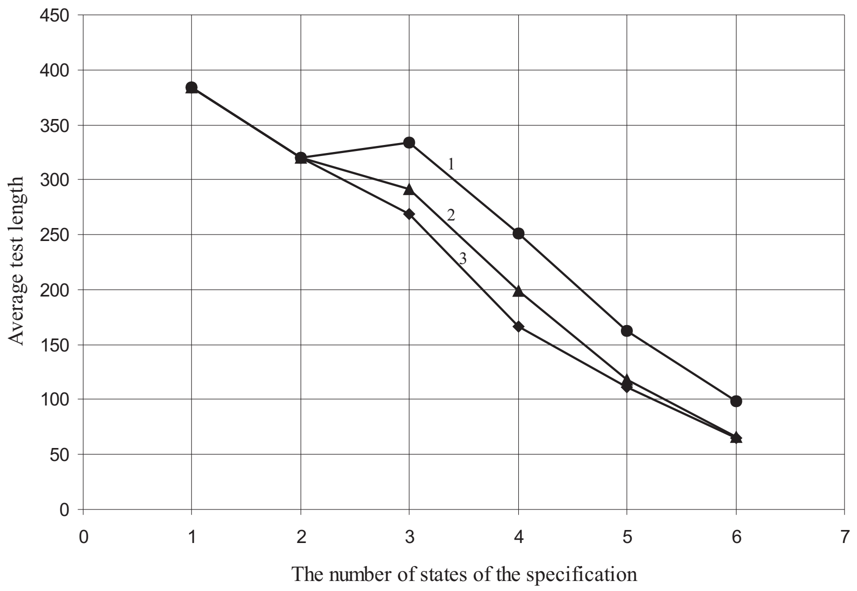

Figure 1. Test length dependence on the number of states of the specification when $m = 6$

1 — the average length of a test suite derived by the W-method
2 — the average length of a test suite derived by the HSI-method
3 — the average length of an enumeration test suite

In Section 4, we analyze the test cases of the shortest m-complete enumeration test.

4. The shortest m-complete enumeration test

Our experiments described in the previous section have shown that the length of an enumeration test exponentially depends on $m - n$ and, thus, is very close to the lengths of the tests derived by the *W*- and *HSI*-methods. Moreover, due to the following theorem, the structure of the enumeration test is the same, i.e., any enumeration test is also based on the set $VPref(X^{m-n+1})$.

Theorem 1. *Any m-complete enumeration test for A contains a subset $\bigcup_{\beta \in V\,\mathrm{Pr}ef(X^{m-n+1})} \beta w_\beta$, where, for any $\beta \in VPref(X^{m-n+1})$, w_β is an identifier of a state where β takes the specification machine A from the initial state.*

Proof. To prove the statement of Theorem 1, we show that for any two states $s, s' \in S$ of the specification machine and for any input sequence $x_1 \dots x_k \in Pref(X^{m-n+1})$ there exists a faulty machine $B \in \Im_m(X, Y)$ such that the shortest sequence distinguishing B from A is $\alpha x_1 \dots x_k \omega$, where α is the shortest sequence taking A from the initial state to the state s, while ω is the shortest sequence distinguishing the state s' from $\delta(s_0, \alpha x_1 \dots x_k)$.

Without loss of generality, we denote the states of A by positive integers $1, 2, \dots, n$ so that the state s is denoted by n. In other words, we consider the specification machine $A = (\{1, 2, \dots, n\}, X, Y, \delta, \lambda, s_0)$ and $\delta(s_0, \alpha) = n$. We denote by s'' the state $\delta(s_0, \alpha x_1 \dots x_k)$ where the sequence $\alpha x_1 \dots x_k$ takes A from its initial state. Then we construct a faulty machine $B = (T, X, Y, \varDelta, \varLambda, t_0)$, where $T = \{1, 2, \dots, n + k - 1\}$ and the functions $\varDelta$ and $\varLambda$ are defined in the following way:

1. $\forall t \in \{1, \dots, n-1\} \forall x \in X\ \varLambda(t, x) = \lambda(t, x)$ and $\varDelta(t, x) = \delta(t, x)$;
2. $\forall t \in \{n, \dots, n+k-2\} \forall x \in X\ \varLambda(t, x) = \lambda(\delta(n, x_1 \dots x_{t-n}), x)$;
   $\forall x \neq x_{t-n+1}\ \varDelta(t, x) = \delta(n, x_1 \dots x_{t-n} x)$; $\varDelta(t, x_{t-n+1}) = t + 1$;
3. $\forall x \in X\ \varLambda(n+k-1, x) = \lambda(\delta(n, x_1 \dots x_{k-1}), x)$;
   $\forall x \neq x_k\ \varDelta(n+k-1, x) = \delta(n, x_1 \dots x_{k-1} x)$; $\varDelta(n+k-1, x_k) = s'$.

Since $k \leq m - n + 1$, it holds that $|T| \leq m$ and, thus, the constructed machine B is in the set $J_m(X)$. Its responses to all sequences that do not traverse the last state $n+k-1$ coincide with those of the specification machine A. The shortest sequence taking B to the state $n + k - 1$ is $\alpha x_1 \dots x_{k-1}$. Under the input x_k, the machine B goes to the state s' while the specification machine A goes to the state $\delta(s_0, \alpha x_1 \dots x_k) = s''$. Finally, since $\lambda(s', \omega) \neq \lambda(s'', \omega)$, the sequence $\alpha x_1 \dots x_k \omega$ distinguishes B from A. Obviously, no shorter sequence possesses this property.

Due to the above considerations, any enumeration test must contain all possible sequences $\alpha x_1 \ldots x_k \omega$ for all states $s \in S$, all sequences

$$x_1 \ldots x_k \in \mathit{Pref}(X^{m-n+1}),$$

and all states $s' \neq \delta(s_0, \alpha x_1 \ldots x_k)$. Thus, it always has a subset

$$\bigcup_{\beta \in V \; \mathit{Pref}(X^{m-n+1})} \beta w_\beta,$$

where w_β is an identifier of a state $\delta(s_0, \beta)$ for any $\beta \in V\mathit{Pref}(X^{m-n+1})$. $\square$

Based on Theorem 1, the lower bound of an m-complete test suite can be established [7]. This lower bound is also shown to depend exponentially on the value of $m - n$ and to deteriorate with increasing n at fixed m.

Due to the above theorem, the explicit enumeration of all faulty machines is of a little help for the derivation of a minimal m-complete test suite. Moreover, the set $V\mathit{Pref}(X^{m-n+1})$, where V is the shortest state cover set of the specification FSM, seems to be the basis of any test derivation method. Therefore, an m-complete test suite can only be shortened based on an appropriate choice and distribution of state identifiers.

In Section 5, we propose a novel modification of the W-method that returns m-complete test suite with the length close to the lower bound.

5. A novel modification of the W-method

Since the set $V\mathit{Pref}(X^{m-n+1})$ is the basis of any test derivation method, identification facilities of the states of the specification FSM are the only thing that can be manipulated in order to shorten the obtained test suite. However, the paper [6] presents an example illustrating that arbitrary state identifiers do not always preserve completeness of a test suite. Below we present the sufficient conditions of m-completeness of a test derived from the set $V\mathit{Pref}(X^{m-n+1})$ and based on the appropriate choice of state identifiers.

Theorem 2. *Let A be a specification FSM, $A = (S, X, Y, \delta, \lambda, s_0)$, $|S| = n$ and V be its prefix-closed state cover set. Consider the set VX^{m-n+1} ($m \geq n$) such that:*

1. $\lambda(\delta(s_0, \alpha_i), w) \neq \lambda(\delta(s_0, \alpha_j), w), \ \forall \alpha_i, \alpha_j \in V$*;*
2. $\lambda(\delta(s_0, \alpha_i), w) \neq \lambda(\delta(s_0, \beta_j), w), \ \forall \alpha_i \in V, \beta_j \in V\mathit{Pref}(X^{m-n+1})$*;*
3. $\lambda(\delta(s_0, \alpha\beta_i), w) \neq \lambda(\delta(s_0, \alpha\beta_j), w), \ \forall \alpha \in V, \beta_i, \beta_j \in \mathit{Pref}(X^{m-n+1})$ *such that β_i is a prefix of β_j.*

The set TS constructed under the above conditions is m-complete test suite for A.

Proof. To prove the statement of the theorem, we consider an arbitrary FSM $B \in J_m(X)$ that is not equivalent to A. There are two cases.

1. FSMs A and B are distinguished by a certain sequence $\alpha \in V$. Then $\alpha \in TS$ and TS distinguishes B from A.
2. No sequence $\alpha \in V$ distinguishes B and A.

In the latter case, consider the sequence $\alpha\beta x,\ \alpha \in V$, that distinguishes B from A and has the shortest suffix βx. If the length of βx is not greater than $m - n + 1$, then $\alpha\beta x$ is a prefix of some sequence of TS, i.e., B is distinguished from A.

Now suppose that the length of βx is greater than $m - n + 1$. Then βx has a prefix β' of length $m - n + 1$ such that $\alpha\beta' \in VPref(X^{m-n+1})$. By construction of the set TS, the set $V \cup \alpha Pref(\beta')$ contains $m + 1$ different sequences. And since the machine B has only m states, two sequences α_1 and α_2 from $V \cup \alpha Pref(\beta')$ take it to the same state t. At the same time, the sequences α_1 and α_2 take the specification to different states s_1 and s_2, since $\alpha\beta x$ is the sequence with the shortest suffix βx distinguishing the two machines. In this case, any sequence $w(s_1, s_2)$ distinguishing the state s_1 from s_2 of the specification distinguishes the state t from either s_1 or s_2. By construction, the set TS has such sequences, i.e. TS distinguishes B from A. Therefore, the set TS is m-complete test suite for the FSM A. □

Due to Theorem 2, below we present an algorithm of an m-complete test suite derivation.

Input. A connected reduced FSM A with n states; a prefix-closed state cover set V; and an integer m greater than or equal to n.

Output. m-complete test suite

The method includes the following steps.

Step 1. Derive the set $TS = VPref(X^{m-n+1})$.

Step 2. For each $\alpha_i \in VPref(X^{m-n+1})$ determine the state s_i where α_i takes the FSM A from the initial state.

For each sequence $\alpha_j \in V \cup Pref(\alpha_i)$, if $\delta(s_0, \alpha_j) \neq s_i$ then check whether TS has sequences $\alpha_j\omega$ and $\alpha_i\omega$ such that ω distinguishes the state $\delta(s_0, \alpha_j)$ from the state s_i.

If there are no such sequences, then include the sequences $\alpha_j\omega$ and $\alpha_i\omega$ into TS for an appropriate distinguishing sequence ω. Repeat this step until all the sequences of the set $VPref(X^{m-n+1})$ are considered. □

The length of the obtained test suite essentially depends on the strategy of selecting the distinguishing sequences ω. We developed an algorithm that allows us to select the best distinguishing sequence according to a special cost function. Since it is our purpose to decrease the length of the obtained test suite, the value of the cost function shows the relative increment in the test length when submitting each distinguishing sequence [10]. The time and space complexity of the proposed algorithm is proportional to the number of pairs of sequences in the set $\mathit{VPref}(X^{m-n+1})$ that are concatenated with distinguishing sequences. The number of sequences in $\mathit{VPref}(X^{m-n+1})$ does not depend on the specification machine and is equal to

$$\frac{|X|^{m-n+1}-1}{|X|-1}+n\cdot|X|^{m-n+1}, \tag{1}$$

where m and n are the numbers of states of the specification and the implementation, respectively, and $|X|$ is the cardinality of the input alphabet. The number of different pairs of states that need to be distinguished according to the proposed algorithm is actually equal to

$$m\cdot n\cdot|X|^{m-n+1}+\sum_{k=0}^{m-n}k\cdot|X|^k-\frac{n\cdot(n-1)}{2}. \tag{2}$$

Thus, the proposed algorithm is enumerative one and, like any other enumeration algorithm, is rather consuming: the time and space consumption is proportional to $\sim m\cdot n\cdot|X|^{m-n+1}$ (at the same time, the upper bound of the test length derived with the W-method [2] is $n^2\cdot m\cdot|X|^{m-n+1}$). Practically, the number of pairs appears to be less than this value, since only different states of the specification need to be distinguished.

Moreover, when $m = n$, the number of states, where the sequences of the set $\mathit{VPref}(X^{m-n+1})$ take the FSM from its initial state, is $n\cdot|X|+1$ and the number of different pairs of states is defined by the value

$$n^2\cdot|X|-\frac{n\cdot(n-1)}{2}(3). \tag{3}$$

Thus, with the help of modern computer systems, it is possible to process FSMs with a rather large number of states.

Example 1. Consider an FSM A in Figure 2.

In our case, $n = 3$ and we consider $m = 4$. By Theorem 2, the set $\{xxx, xyxx, yxxx, yxyxx, yyxxx, yyxxy, yyxyy, yyyxx, yyyyx, yyyyy\}$ is a 4-complete test suite. By direct inspection, one can see that the state identifiers y and x that are used to check the transition under y after the sequence $yyx \in VX^{m-n}$ and the transition under x after the sequence $yyy \in VX^{m-n}$

A	**1**	**2**	**3**
x	**1**/0	**2**/1	**1**/1
y	**3**/1	**3**/1	**2**/0

Figure 2

are not harmonized. The same situation holds for the pair of sequences *yyxy* and *yxx* taking the specification to states 3 and 1, respectively, and for the pair of sequences *yyxy* and *xx*.

Keeping in mind that the reset input is submitted at the beginning of each test case, the total length of the test suite is 46+10=56.

At the same time, the *HSI*-method returns a 4-complete test suite {*xxx*, *xyx*, *xyy*, *yxxx*, *yxyx*, *yxyy*, *yyxxx*, *yyxxy*, *yyxyx*, *yyxyy*, *yyyxx*, *yyyyx*, *yyyyy*} for the family of harmonized state identifiers $\{W_1, W_2, W_3\}$, where $W_1 = \{x\}$, $W_2 = \{x, y\}$, $W_3 = \{y\}$. The total length of the test suite is equal to $56 + 13 = 69$.

Figures 3 and 4 represent the results of computer experiments. In Figure 3, the average length of a 10-complete test suite derived by the *HSI*-method and its proposed modification are compared with the lower bound. Figure 4 represents the average length of a test suite when $m = n$ for $n = 2, \ldots 10$.

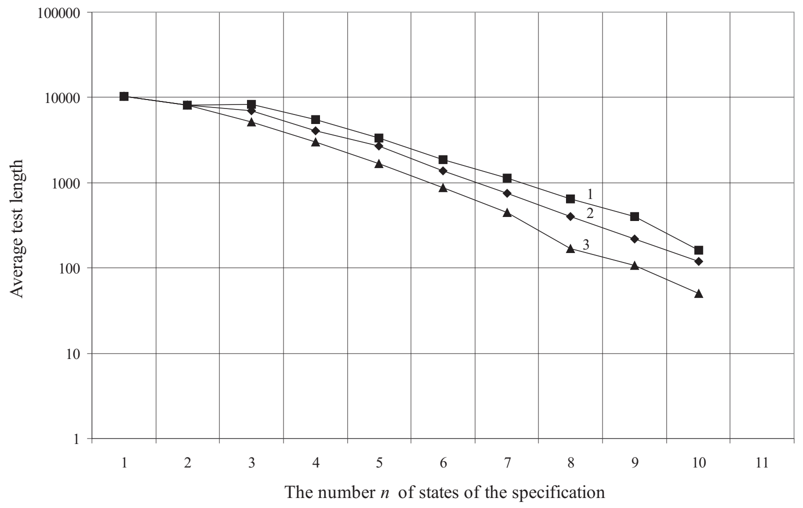

Figure 3. The test length dependence on the number of states of the specification when $m = 10$

1 — the average length of the test suite derived by the *HSI*-method
2 — the average length of the test suite derived by the proposed modification of the W-method
3 — the lower bound of a 10-complete test suite

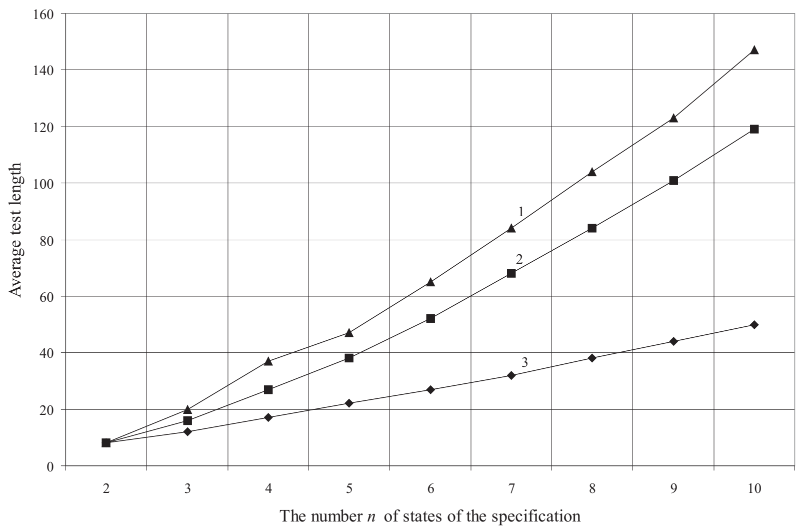

Figure 4. The average length of the tests derived by different methods when $n = m$

1 — the average length of the test suite derived by the *HSI*-method
2 — the test length derived by a new modification of W-method
3 — the lower bound of the m-complete test suite, $m = 2, ..., 10$ [7]

The experiments clearly show that the method proposed in this paper returns the tests shorter than other existing methods. Additionally, time consumption of the proposed method is comparable with time consumption of the *HSI*-method and is very close to that of the W-method [10].

6. Conclusion

In the paper, we have studied a possibility to reduce the total length of the m-complete test suite derived from a given specification FSM. We show that the total length exponentially depends on the difference between m and the number of states of the specification FSM and can be only reduced by an appropriate choice and distribution of state identifiers. Based on this observation, we have proposed a novel modification of the W-method. The performed experiments clearly show that the proposed modification returns the m-complete test suite shorter than other methods. Our future work is directed to develop the strategies of how to select and distribute the state identifiers in order to obtain the shortest test suite.

References

[1] Pomeranz I., Reddy S. M. Test generation for multiple state-table faults in finite-state machines // IEEE Trans. Computers. — 1997. — Vol. 48, N 7. — P. 783–794.

[2] Vasilevsky M. P. Failure diagnosis of automata // Cybernetics, Plenum Publishing Corporation, N.Y. — 1973. — N 4. — P. 653–665.

[3] Chow T.S. Test software design modeled by finite state machines // IEEE Trans. Software Eng. — 1978. — Vol. SE-4, N 3. —P. 178–187.

[4] Fujiwara S., Bochmann G. v., Khendek F., Amalou M., Ghedamsi A. Test selection based on finite state models // IEEE Trans. Software Eng. — 1991. — Vol. SE-17, N 6. — P. 591–603.

[5] Petrenko A., Yevtushenko N., Bochmann G. V. Testing deterministic implementations from their nondeterministic specifications // Testing of Communicating Systems: Proc. / IFIP 9th Intern. Workshop, Germany, 1996. — Germany, 1996. — P. 125–140.

[6] Wendy Y. L. Chan, Son T. Vuong, Robert Ito M. An improved protocol test generation procedure based on UIOS // ACM Trans. Computer Systems — 1989. — P. 283–294.

[7] Koufareva I., Kossareva M. Evaluating length of an enumeration test // Theory and Technique of Information Transmitting and Processing: Proc. / 7th Intern. Conf., Charkov, Ukraine, Oct. 1–4, 2001. — Charkov, 2001. — P. 299–300 (in Russian).

[8] Petrenko A., Yevtushenko N. Test Derivation from Partial Specifications. — Kluwer Academic Publishers, 2000.

[9] Koufareva I., Kossareva M. A study of FSM-based test derivation strategies // ICADM'2000: Proc. / 3d Russian Conf. in Applied Discrete Mathematics, Tomsk, Russia, Sept. 10–12, 2000. — Tomsk, 2000. — P. 204–209 (in Russian).

[10] Dorofeeva M., Koufareva I. FSM-based strategies of test suite minimization // ICADM'2002: Proc. / 4th Russian Conf. in Applied Discrete Mathematics, Tomsk, Russia, Sept. 10–13, 2002. — Tomsk, 2002. — P. 357–363 (in Russian).