Front. Electr. Electron. Eng. China 2007, 2(1): 29–33
DOI 10.1007/s11460-007-0005-z

RESEARCH ARTICLE

LIN Huahui, ZHAO Baohua, QU Yugui

Generation method of minimal-complete-coverage interoperability test sequence based on digraph

Abstract Even if two implementations of a protocol pass the conformance testing, it cannot guarantee that they can interoperate properly; so direct testing of interoperation is considered indispensable. During the interoperability testing, a minimal number of test sequences are expected to check as many as possible implementation errors. By using minimal-complete-coverage criterion, the test sequence generation based on digraph can produce more effective test sequences.

Keywords interoperability test, interoperability equivalence, minimal-complete-coverage criterion, algorithm of test sequence generation, digraph

1 Introduction

Protocol conformance testing is to check whether a protocol implementation is in conformance with its specification [1], while interoperability testing is to check whether two or more protocol entities can interact correctly with each other and accomplish the expectant task [2]. Conformance testing cannot ensure protocol entities' interoperability [3], so to apply interoperability testing directly to protocol implementation is indispensable. Since the cost to apply test in real environment is high, how to design an automatic test suit generating algorithm to cover more implementation errors with less test cases has become an important task in research area in interoperability testing. Coverage is the most important guideline to judge a test suit generating algorithm [4]. When we discuss the coverage criteria of interoperability, we always assume that interoperability errors occur only when the interaction happens between systems. Complete-coverage testing [2] can test the interoperation between systems completely, but redundant tests are contained in the test suit generated by this algorithm. In this paper, we bring forward stricter interoperation test coverage criteria: minimal-complete-coverage criteria. We also give a relative test suit generating algorithm, which can generate test suit with minimal test cases number and the same error-checked ability with complete-coverage test suit.

Translated from *Journal of University of Science and Technology of China*, 2006, 36(2): 225−229 [译自: 中国科学技术大学学报]

LIN Huahui, ZHAO Baohua (✉), QU Yugui
Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China
E-mail: bhzhao@ustc.edu.cn

2 Principle of interoperability test

2.1 Base framework of interoperability test

Interoperability test is to check whether more than one implementation under test (IUT) can interact correctly, and the base test framework is shown in Fig. 1.

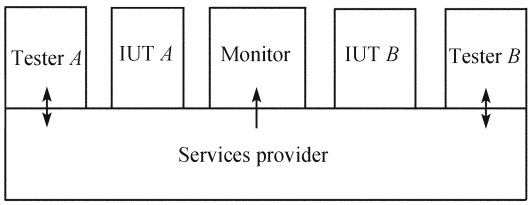

Fig. 1 Base framework of interoperability testing

Tester *A* and tester *B* send messages to IUT *A* and IUT *B*, respectively, which will result in *A* and *B*'s interoperation, and the monitor observes interactions between two IUTs and checks whether they conform to expectation. Different test frameworks will be used to relate to the actual environment and the difference of testers' controllability and observability of IUT [5,6].

2.2 Protocol formalization model

Finite state machine (FSM) model is usually used to describe the control behavior of protocol in protocol engineering research area, and test suit generating algorithms are discussed based on it. FSM is defined as follows.

Definition 1 FSM: an FSM M is a 5-tuple (S, s_0, I, O, T), S is the state set, s_0 is the initial state, $s_0 \in S$; I and O are input

set and output set, respectively; T is the transition set, each transition t in T can be denoted as $(s, s', x/y)$; $s \in S$ is the head-state of t and $s' \in S$ is the tail-state, x/y is the input/output when transition occurs, $x \in I$, $y \in O$.

When two entities communicate with each other, we can use composite FSM to describe their global behaviors. Composite finite state machine can be obtained by reachability analysis of each communicating entity. Composite FSM is defined as follows:

Definition 2 Composite FSM: the composite FSM M of FSM A and FSM B is a 5-tuple $(S^2, s_0^2, I^2, O^2, T^2)$. $S^2 = \{s^2_0, s^2_1, \ldots, s^2_{n-1}\}$ is the state set of M; $s_i^2 = (s_{a_i}, s_{b_j})$ is a global state composed of A's state s_{a_i} and B's state s_{b_j}; $s^2_0 = (s_{a_0}, s_{b_0}) \in S^2$ is the initial state; $I^2 = \{v_1^2, v_2^2, \ldots, v_m^2\}$ is called the input notation set; $v_i^2 \in I_a \cup I_b - I_a \cap O_b - O_a \cap I_b$ is the external input notation; $O^2 = \{u_1^2, u_2^2, \ldots, u_k^2\}$ is the output notation set; $u_i^2 = \langle o_1, o_2, \ldots, o_p \rangle$ is the output notation sequence; $o_i \in O_a \cup O_b$, $T^2 = \{t_1^2, t_2^2, \ldots, t_j^2\}$ is the transition set, and each transition t_i^2 can be denoted as $\{s^2, s^{2'}, v/u\}$, $s^2, s^{2'} \in S^2$, $v \in I^2 \in u \in O^2$.

In composite FSM, the messages exchanged between A and B are called internal messages and others are called external messages.

To generate interoperability test sequences, first we can construct the composite FSM of two IUTs, and then use some test sequences generating algorithm to generate its test sequences base on the composite FSM.

Definition 3 Local transition: in composite FSM, the transition without internal messages is called local transition. For example, $t_i^2 = \{s^2, s^{2'}, v/u\}$, if u does not contain internal messages, then t_i^2 is local transition.

Definition 4 Interoperating transition: in composite FSM, the transition with internal messages is called interoperating transition.

Definition 5 Interoperating test sequence (ITS): interoperating test sequence TS $= \langle t_0^2, t_1^2, \ldots, t_m^2 \rangle$ is a continuous state-transition sequence in composite FSM, where $t_i^2 \in T^2$, and the head-state of t_0^2 is the initial state s_0^2.

FSM can be represented by digraph $G = (E, V)$, where each directed edge e in edges set E represents a transition in FSM, and each vertex v in vertexes set V represents a state. An interoperating test sequence is denoted by a sequential edge set, which is a trace in digraph.

2.3 Base strategy of ITS generating method base on digraph

To a composite FSM, which is represented by digraph G, its ITS can be generated with the following steps [2].

Step 1 Find out all the strong-connected components (SCCs) in digraph G.

Step 2 Represent each SCC by a vertex and construct a directed acyclic graph (DAG) G_1.

Step 3 Find out all acyclic paths from source vertex to sink vertex in G_1.

Step 4 Find out all acyclic paths and simple path in each SCC.

Step 5 Represent the vertex in Step 3 with an acyclic path in relative SCC to obtain the acyclic path in G.

Step 6 Compose all acyclic paths in Step 5 and all simple paths in Step 4 and generate the final path set, which represents the interoperability test suite of the composite FSM.

If the original digraph is dealt with by different strategies, we can generate some test sets that satisfy different coverage criteria. To obtain the optimal test set, we will discuss the coverage criteria of interoperability. Minimal-complete-coverage criteria is defined and a relative test sequence generating algorithm is given.

3 Minimal-complete-coverage criteria of interoperability test

As is supposed in Ref. [7], interoperability errors occur only when interaction between systems happens, so only interoperating transitions need to be considered when test sequence is generated. All interoperability errors can be checked so long as the test sequences in the test suit can cover all interoperating transitions between IUTs. The local transition is only used to lead to the interoperating transition. If two test sequences contain the same subinteroperating test sequences, we will consider that they have the same error-checking ability in interoperability test.

Definition 6 Interoperating equivalence test sequence: for two ITS, TS1 and TS2, if the subsequence constructed by all interoperating transitions from TS1 is similar to that from TS2, then T_1 and T_2 are called the equivalent test sequences, denoted as $T_1 \leftrightarrow T_2$.

For example, $T_1 = \langle tl_1, tl_2, ti_1, tl_3, ti_2 \rangle$, $T_2 = \langle tl_1, tl_2, ti_1, tl_4, tl_5, ti_2 \rangle$, ti_1 and ti_2 are interoperating transitions, others are local transitions, then $T_1 \leftrightarrow T_2$. Two test sequences cover the interoperating transition sequence $\langle ti_1, ti_2 \rangle$, so when test case is generated, only one is enough.

Suppose that in composite FSM M, TS0 represents exhaust-coverage test set [2], $|T|$ is the length of test sequence T, is the interoperating transition set, then the minimal-complete-coverage interoperability test criteria can be defined as follows:

Definition 7 Minimal-complete-coverage interoperability test criteria: if a test set (TS) of composite FSM M_1 satisfies the following three conditions, then we say that the TS satisfies the minimal-complete-coverage interoperability test criteria.

1) $\forall T \in \text{TS0}, \exists T' \in \text{TS} : T' \leftrightarrow T \wedge |T'| \leqslant |T|$;

2) $\forall T \in \text{TS}, \rightharpoonup \exists T' \in \text{TS} : T' \leftrightarrow T$;

3) $\forall T \in \text{TS}, \exists t \in T : t \in \text{TrIS}$.

It is easy to see that the test set that satisfies the minimal-complete-coverage interoperability test criteria is a minimal set that covers all possible interoperating transitions.

4 Minimal-complete-coverage interoperability test sequence generating algorithm

When interoperability test sequences are generated, if two interoperating transitions can be concatenated by more than one local transition subsequence, for example, in test sequences $\langle ti_1, p_1, ti_2\rangle$ and $\langle ti_1, p_2, ti_2\rangle$, p_1 and p_2 are two subsequences that only contain local transition, ti_1 and ti_2 are interoperating transitions, only the shortest local transition subsequence needs to be chosen and test sequences for other local transitions are not necessary. So before the test sequences generation, we can use an edge, which represents the shortest local transition subsequences between two interoperating transitions and called superedge, to replace all local transition subsequences between the two interoperating transitions and then generate a new digraph. After that, we apply exhaust-coverage test sequences generating algorithm to the new digraph and obtain the test sequences set. For each test sequence in the set, replace the superedge with its relative shortest local transition subsequence, then we can get the final test set, which satisfies the minimal-complete-coverage interoperability test criteria.

One data structure and two relative algorithms are introduced from Ref. [2].

Data structure: next-transition-tree $T(v)$: $T(v)$ is to save the acyclic path from v to all other vertexes in an SCC.

Algorithm 1 Path-in-DAG

Input: a DAG G with one source node and more than one sink node

Output: all acyclic paths from source node to sink nodes in G

1) Sort vertexes in G in topological order: $s = v_0, v_1, \ldots, v_{n-1}$.

2) Let i from $n-1$ to 0: if v_i is the terminating vertex, then let $p(v_i) = \{\}$, else for each outgoing edge $w_1, w_2, \ldots, w_r$ of v_i, $p(v_i) = (v_1, w_1)p(w_j) \cup (v_1, w_2)p(w_2) \cup \ldots \cup (v_1, w_r)p(w_r)$.

3) Return $p(v_0)$.

Algorithm 2 Exhaust-coverage

Input: digraph G for composite FSM with designated source node

Output: the minimal acyclic path set P, which satisfies exhaust-coverage criteria

1) Find out all SCCs in G.

2) For each SCC: construct in the next-generation-tree of each vertex.

3) Shrink each SCC into a vertex and generate a new DAG G'.

4) Apply path-in-DAG algorithm to G' and then generate the acyclic path set P', which includes all acyclic paths from source vertex to sink vertex.

5) For each path p' in P': replace the vertex that shrinks from SCC with its relative acyclic path and generates the acyclic path p in digraph G, thus constructing G's acyclic path set P.

According to the above discussion, we can construct the test sequences generating algorithm satisfying the minimal-complete-coverage interoperability test criteria.

Algorithm 3 Minimal-complete-coverage

Input: digraph G for composite FSM with designated source node

Output: the minimal acyclic path set P, which satisfies minimal-complete-coverage criteria

1) For source node v_0, find out vertex set U_0 containing all start vertexes of black edges, which is available from white edge; for each vertex u in U_0, add superedge $e = (v_0, u)$, and find out the shortest white-age sequence from v_0 to u, which is denoted as WP(e).

2) For each end vertex v of black edge, find out the vertex set U. U contains all start vertexes and end vertexes of black edges, which are reachable from v. For each vertex u, add a superedge $e = (v, u)$ into G and find out the shortest white edge sequence denoted as WP(e).

3) Delete all white edges in G and eliminate all connerity sets without source node, then generate new digraph G'. Apply exhaust-coverage algorithm to G' and obtain the acyclic path set P', eliminate all paths that contain two continuous superedges.

4) For each path p' in P': use WP(e) to replace each superedge e in p' and generate path p of original digraph G, thus constructing the acyclic path set P of G.

5) Eliminate the potential loop of each path in P.

Algorithm 4 Minimal-complete-test

Input: composite FSM M

Output: interoperability test sequences set TS, which satisfies minimal-complete-coverage criteria

1) Construct the digraph G for M.

2) Apply minimal-complete-coverage algorithm to G and obtain the acyclic path set P.

3) Construct the simple cyclic path set C. C contains all simple cyclic paths, which include at least one black edge in G.

4) Compose P and C and then generate the final test set TS.

The relationships between the above algorithms are: Algorithm 4 calls Algorithm 3, Algorithm 3 calls Algorithm 2, and Algorithm 2 calls Algorithm 1. We will take an example to demonstrate the process of the algorithms.

For a designated composite FSM M, first we apply step 1) of Algorithm 4 to generate a digraph as is shown in Fig. 2(a). In Fig. 2(a). tl_1–tl_{10} are local transitions, ti_1–ti_3 are interoperating transitions. Then Algorithm 4 calls Algorithm 3 to obtain the acyclic path P. After step 1), and step 2) of Algorithm 3 are applied, superedge s_1–s_6 will be added into digraph, when WP(s_1) = $\langle tl_1\rangle$, WP(s_2) = $\langle tl_2\rangle$, WP(s_3) = $\langle tl_5, tl_{10}\rangle$, WP($s_4$) = $\langle tl_8\rangle$, WP(s_5) = $\langle tl_5, tl_8\rangle$, WP($s_6$) = $\langle tl_{10}\rangle$, then Fig. 2(b) is generated. Step 3) deletes all white edges and then eliminates the unreachable vertexes (6, 7, 9), then generates Fig. 2(c). Step 4) generates the path set P' of Fig. 2(c), where $P' = \{\langle s_1, ti_1, s_3\rangle, \langle s_1, ti_1, s_5, ti_3\rangle, \langle s_2, ti_2, s_6\rangle, \langle s_2, ti_2, s_4, ti_3\rangle\}$. Step 5) replaces superedge e in P' with WP(e),

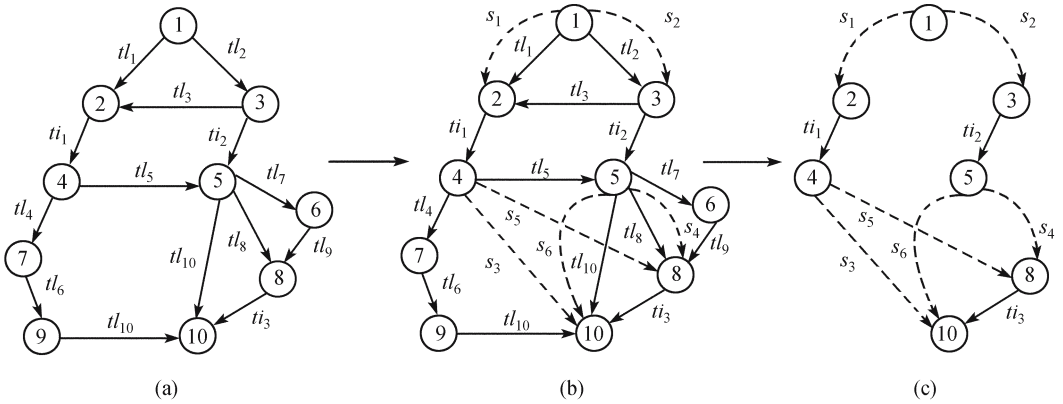

Fig. 2 Process of minimal-complete-coverage interoperability test sequence generation

then obtains $P = \{\langle tl_1, ti_1, tl_5, tl_{10}\rangle, \langle tl_1, ti_1, tl_5, tl_8, ti_3\rangle, \langle tl_2, ti_2, tl_{10}\rangle, \langle tl_2, ti_2, tl_8, ti_3\rangle\}$. All paths in P are acyclic. After Algorithm 3 is finished, Algorithm 4 continues to perform step 3) and step 4) and generates the final test sequences set P.

In contrast, we also apply complete coverage algorithm [2] to Fig. 2(a), and obtain test sequences set P_1. $P_1 = \{\langle tl_1, ti_1, tl_5, tl_{10}\rangle, \langle tl_1, ti_1, tl_5, tl_8, ti_3\rangle, \langle tl_2, ti_2, tl_{10}\rangle, \langle tl_2, ti_2, tl_8, ti_3\rangle, \langle tl_1, ti_1, tl_5, tl_7, tl_9, ti_3\rangle, \langle tl_2, ti_2, tl_7, tl_9, ti_3\rangle\}$, where $\langle tl_1, ti_1, tl_5, tl_8, ti_3\rangle$ and $\langle tl_1, ti_1, tl_5, tl_7, tl_9, ti_3\rangle$ are interoperating equivalents, the same with $\langle tl_2, ti_2, tl_8, ti_3\rangle$ and $\langle tl_2, ti_2, tl_7, tl_9, ti_3\rangle$.

We can see from the above result that the test sequences set generated by minimal-complete-test algorithm is smaller than the complete coverage algorithm, but interoperating-error-check ability will not be reduced. We will confirm these with some experimental data from a real system in the next chapter.

Next, we will discuss the correctness of the algorithm curtly: it only needs to illustrate that the test sequences generated by the algorithm satisfy the minimal-complete-coverage criteria.

Step 1) of Algorithm 3 guarantees that no test sequence with pure local transitions will be contained in the test sequences set, which are generated by the algorithm, that is, the criteria condition 3) is satisfied. Reduction to absurdity method can be used to describe condition 2). Any two sequences in test sequence sets are different. Suppose that there are two interoperating equivalence sequences in one set, and then there will be two different local transition subsequences between two interoperating transitions to a certainty. But step 1) and step 5) in Algorithm 3 can ensure that in any test sequence, two interoperating transitions will be concatenated by only one superedge, which will be replaced by the relatively shortest local transitions sequence. So there will not exist such two different local transition subsequences. This is not confirmed, i.e., the algorithm will not generate two interoperating equivalence sequences.

Consider condition 1), exhaust coverage test algorithm can cover all test sequences of composite FSM. In contrast to exhaust coverage test, minimal-complete-coverage test only optimizes the local transition in test sequences and as the superedge is introduced, the connexity of interoperating transitions will not be changed. So if there is an interoperating transition subsequence in the exhaust coverage test, the same subsequence can be found in the minimal-complete-test set. And in step 5) of Algorithm 3, the superedge will be replaced, but the shortest local transition subsequence can ensure that all test sequences in the test set are shortest.

5 Result of experiment

We used the algorithm to generate test suit for the interoperability test of traditional POTS phone call based on IP network of a VoIP system [2]. The composite FSM of VoIP end-user contains 21 states, 24 interoperability transitions, and 44 local transitions. 252 test sequences are generated, but 590 test sequences will be generated if we use complete coverage test algorithm [2]. We can see that our algorithms are more optimal. When the number of local transitions becomes larger in composite FSM, our algorithm can reduce the test set evidently without hurting the coverage ability.

6 Conclusions

In this paper, we bring forward a new test criteria in interoperability test called minimal-complete-coverage criteria, and give test sequences generating algorithms that satisfy the criteria. The test set generated by this algorithm is a minimal set that can cover all interoperating transition sequences. Thus we can check more interoperating errors by executing fewer test sequences. The experimental result shows that the algorithm can be applied to the real system as well.

Acknowledgements This paper was supported by the National Natural Science Foundation of China (Grant Nos. 60241004, 60602016 and 60602016), the National Basic Research Program of China (Grant No. 2003CB314801), HUAWEI Foundation (No. YJCB2006044TS).

References

1. ISO/IEC9646-1. Information technology: Open system interconnection: Conformance testing methodology and framework: Part 1: General concepts, 1994, 1−20
2. Hao Ruibing. Integrated system interoperability testing with applications to VoIP. In: Proceedings of the IFIP TC6 WG6.1 Joint Inter. Conf. FORTE XIII/PSTV XX, 2000, 69−84
3. Rafiq O, Castanet R. From conformance testing to interoperability testing. In: Proceedings of the 3rd International Workshop on Protocol Test System. North-Holland, Amsterdam. 1990, 371−385
4. Lu Xinyan, Zhou Hao, Zhao Baohua. A dynamic protocol conformance testing method. Journal of University of Science and Technology of China, 2005, 35(3): 338−404 (in Chinese)
5. Shin J, Kang S. Interoperability test suite derivation for the ATM/B-ISDN signaling protocol. Testing of Communicating Systems, 1998, 11: 313−330
6. Vadim T. Interoperability testing based on a fault model for a system of communicating FSMs. Testing of Communicating Systems, 2003, 226−242
7. Kang Sungwan. Interoperability test suite derivation for symmetric communication protocols. In: Proceedings of FORTE/PSTV'97. Osaka, Japan. 1997, 57−72