論文 97-22-2-4

On the Generation of Synchronizable Conformance Test Sequences Using the Duplex Digraph and Distinguishing Sequences

Chul Kim*, Joo Seok Song** *Regular Members*

이중 방향그래프와 구별시퀀스를 이용한 동기적 적합성시험 항목의 생성

正會員 김 철*, 송 주 석**

Abstract

In this paper, a new technique is proposed for generating a minimum-length synchronizable test sequence that can be applied in the distributed test architecture where both external synchronization and input/output operation costs are taken into consideration. The method defines a set of transformation rules that constructs a duplex digraph from a given finite state machine representation of a protocol specification such that a rural Chinese postman tour of the duplex digraph can be used to generate a minimum-length synchronizable test sequence using synchronizable distinguishing sequences as the state identification sequence for each state of the given finite state machine. This method provides an elegant solution to the synchronization problem that arises during the application of a predetermined test sequence in some protocol test architectures that utilize remote testers.

요 약

본 논문은 통신 프로토콜의 적합성 시험시 외부 동기와 입출력 시험 항목의 비용이 고려되어야 할 분산 시험 환경에서 적용될 수 있는 동기적 시험 항목 생성을 위한 새로운 방법을 제시한다. 이 방법은 유한 상태 기계로 표현된 프로토콜의 명세로 부터 이중 방향 그래프를 구성한 후 최소 길이의 동기적 시험 항목을 생성하기 위해 rural Chinese postman tour(RCPT) 알고리듬을 적용한다. 또한 시험시 주어진 유한 상태 기계의 각 상태에 대한 상태 식별 항목으로써 synchronizable distinguishing sequence(SDS)들을 사용한다. 본 논문의 방법은 원거리 시험기들을

*용인대학교 전산통계학과
Dept. of Computer Science & Statistics, Yongin University
**연세대학교 컴퓨터과학과
Dept. of Computer Science, Yonsei University
論文番號:96285-0906
接受日字:1996年 9月 6日

사용하는 프로토콜 시험 구조에서 기 결정된 시험 항목을 적용할 때 시험 대상과 이들 시험기들간에 발생할 수 있는 동기화 문제에 대한 해결책으로 제시될 수 있다.

I. Introduction

A communication protocol defines all possible interactions among the communicating entities. Conformance testing is to establish whether a protocol implementation under test (IUT) conforms to the protocol specification [1]. When conformance testing is performed, an IUT is viewed as a black box. The internal behavior of the IUT cannot be observed, but only the output sequences can be checked after applying the input sequences to the IUT. In Fig. 1, the lower interface and the upper interface of the IUT are controlled and observed indirectly by the lower tester (LT) and directly by the upper tester (UT), respectively. The LT and the UT coordinate to stimulate the IUT with a given sequence of input interactions and to observe the resulting output from the IUT.

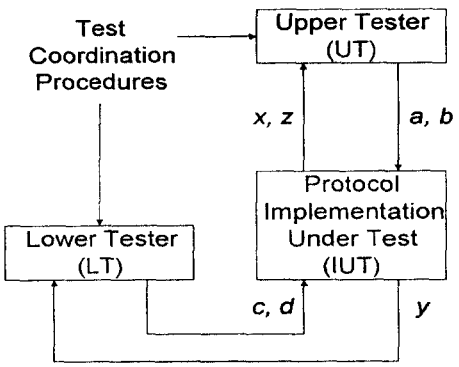

Fig. 1 The Distributed Test Architecture for Testing a Protocol Implementation.

During the application of a predetermined test sequence in the test architecture, the UT and the LT are bound to synchronize with each other only through their interactions with the IUT. However, this requirement may lead to a synchronization problem when the LT (or the UT) is expected to send an input to the IUT after the IUT sends an output to the UT (or the LT) but the LT (or the UT) is unable to determine whether the IUT sent that output. Synchronization between the UT and the LT can be achieved by external synchronization operations. However, these solutions either require an additional communication channel and/or an additional protocol for coordination between the UT and the LT. Such requirements can be eliminated by constructing a synchronizable test sequence such that the corresponding sequence of transitions cause no synchronization problem.

In [2], the technique can only be applied to the protocol which is tightly synchronizable with both the UT and LT. The duplex digraph of the other protocol is not strongly connected, of which no tour can be employed to generate a test sequence to completely test all the transitions. In [3], synchronization problem must be considered as part of the architectural constraints and design of test systems. However, it is not clear whether the synchronization problem can always be avoided by making appropriate changes to the protocol specification as mentioned in the paper. Therefore, the method will only be applicable to a limited extent in the case of a real protocol. In [4], the method constructs a test sequence using synchronizable unique input/output sequences for each state of the given finite state machine. But, a recent research [5] mentions two general classes of faults-nonminimal implementation and incomplete specification-that cannot be caught with an unique input/output based testing. Therefore, all the previous researches we reviewed have their own pros and cons.

This paper presents a new technique for generating a minimum-length synchronizable test sequence that can be applied in the distributed test architecture where the cost of both external synchronization

operations and input/output operations are taken into consideration. The main advantage of this technique is that it ensures complete fault coverage. The rest of the paper is organized as follows. Section 2 describes some notations and concepts related to an FSM model and its testing. Section 3 and 4 present the synchronization problem and the duplex technique [2, 4], respectively. Section 5 introduces a set of transformation rules that constructs a duplex digraph from a given FSM representation of a protocol specification such that a rural Chinese postman tour [6] of the duplex digraph can be used to generate a minimum-length synchronizable test sequence using synchronizable distinguishing sequences. Finally, a conclusion is given in Section 6.

Ⅱ. FSM Model and Testing

The control portion of a protocol can be specified as an FSM M with a quintuple (S, I, O, f, g), where:

S = the set of states of M, including a special state s_1 called the initial state;
I = the set of inputs, written i in the following, $i_p \in I$;
O = the set of outputs, written o in the following, including the null output (nu), $o_q \in O$;
f = the next-state (transition) function, $S \times I \rightarrow S$;
g = the output function, $S \times I \rightarrow O$.

An FSM M is represented as a directed graph, $G=(V, E)$, where the set of vertices $V=\{v_1,...,v_n\}$ represents the set of specified states $S=\{s_1,...,s_n\}$ of M and a directed edge $(T_m; v_j, v_k; i_p/o_q) \in E$ represents a transition from state s_j to state s_k in M. Each edge has a label i_p/o_q where i_p and o_q are input and output operations.

The process of checking a transition from sj to sk with input/output i_p/o_q consists of three basic steps:

(1) The FSM implementation is put into state s_j;
(2) Input i_p is applied and the output is checked to verify that it is o_q, as expected; 0
(3) The new state of the FSM implementation is checked to verify that it is s_k, as expected.

The cost (or length) of each edge of G is equal to the number of input/output pairs in its label. The cost (or length) of a path in G is the sum of the costs (or lengths) of edges included in the path. A path with the minimum-cost (or-length) among all paths from v_j to v_k is called a shortest path from v_j to v_k. $G=(V, E)$ is a subgraph of $G^*=(V^*, E^*)$ if $V \subseteq V^*$ and $E \subseteq E^*$. An edge-induced subgraph $G[E_c]$ of E^* is the subgraph of G* whose vertex set is the set of heads and tails of edges in E_c, and whose edge set is E_c where $E_c \subseteq E^*$ [6]. A tour in G is a path which starts and ends at the same vertex. An Euler tour P of G is a tour which contains every edge in E exactly once. A rural Chinese postman tour (RCPT) T of $G=(V, E)$ over a set $E_c \subseteq E$ is a minimum-cost (or-length) tour traversing every edge in E_c at least once [6].

An example of a graph representation and its transition table of an FSM M are shown in Fig. 2 and Table 1, respectively. An input (or output) operation of an IUT is said to be related to LT if it comes from (or goes to) the lower tester LT. An input (or output)

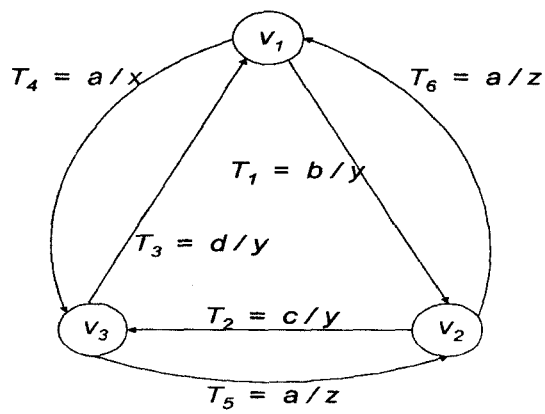

Fig. 2. A Graph Representation of a Finite State Machine M.

operation of M is said to be related to UT if it comes from (or goes to) the upper tester UT. For example, in the FSM in Fig. 2, the symbols c, d (or y) refer to an operation related to LT (and not related to UT) and the symbols a, b (or x, z) refer to an operation related to UT (and not related to LT).

Table 1. Transition Table for M in Fig. 2

	Output				Next-State			
Input / State	a	b	c	d	a	b	c	d
v_1	x	y	nu	nu	v_3	v_2	v_1	v_1
v_2	z	nu	y	nu	v_1	v_2	v_3	v_2
v_3	z	nu	nu	y	v_2	v_3	v_3	v_1

Ⅲ. Synchronization Problem

For most of conformance test generation methods [7,8] reported in the literature it was implicitly assumed that all interactions of the IUT are directly visible to the tester. However, this is not always true in the case of protocol testing as shown in Fig. 1, which shows the distributed test architecture [1,9] for OSI conformance testing. In this architecture the upper and lower testers see only the interactions taking place on the upper and lower interface of the protocol entity, respectively. The synchronization between the upper and lower testers is in general a problem. The synchronization problem is defined as follows:

Definition 1:

A pair of consecutive transitions whose labels form a sequence of input/output operations $[i_1/o_1, i_2/o_2]$ encounters a *LT to UT synchronization problem* if both i_1 and o_1 are related to LT (and not to UT), where i_2 is related to UT; it encounters a *UT to LT synchronization problem* if both i_1 and o_1 are related to UT (and not to LT), where i_2 is related to LT.

For an example as shown in Fig. 2, the sequence of input/output operations $[c/y, a/z]$ (generated from the pair of transitions $[T_2, T_5]$) encounters a *LT to UT synchronization problem*. In $[c/y]$, LT will send c to IUT and then receive y from IUT, while in $[a/z]$, UT is expected to send a to IUT; however, UT can not detect the time when IUT receives c and therefore cannot determine the time to send message a to IUT. In order to solve the synchronization problem, two external synchronization operations *LT to UT* and *UT to LT* are proposed by ISO 9646 [7] as follows:

Definition 2:

LT to UT: LT informs UT that it is now a right time to send the next message.

UT to LT: UT informs LT that it is now a right time to send the next message.

The *LT to UT* (or *UT to LT*) external synchronization operation is introduced each time the *LT to UT* (or *UT to LT*) *synchronization problem* is encountered. In the below, a synchronizable test sequence which involves external synchronization operations is defined to be a sequence of input/output operations that do not encounter the synchronization problem as follows:

Definition 3:

A test sequence $[i_1/o_1, X_1, i_2/o_2, X_2, \ldots, X_{n-1}, i_n/o_n]$ is synchronizable if, for $1 \le k \le n-1$, either

(1) $[i_k/o_k, i_{k+1}/oi_{k+1}]$ encounters no synchronization problem and Xk is null, or

(2) $[i_k/o_k, i_{k+1}/o_{k+1}]$ encounters the *LT to UT synchronization problem* and X_k is an external synchronization operation *LT to UT*, or

(3) $[i_k/o_k, i_{k+1}/o_{k+1}]$ encounters the *UT to LT synchronization problem* and X_k is an external synchronization operation *UT to LT*.

For example in the transition digraph of FSM M in Fig. 2, the test sequence $[c/y, a/z]$ generated from the pair of transitions $[T_2, T_5]$ encounters a synchronization problem. This test sequence can be converted into a synchronizable test sequence $[c/y, LT\ to\ UT, a/z]$ which involves the *LT to UT* external synchronization

operation. Obviously, the resulting sequence is synchronizable because the above condition (2) is satisfied.

Ⅳ. The Duplex Technique

The duplex technique [2] is proposed such that its shortest paths can be used to generate minimum-cost (or -length) synchronizable transfer sequences, where both the input/output and external synchronization operation costs are taken into consideration. The graph $G(V, E)$ of an FSM M is converted into the duplex digraph $G'(V', E')$ as shown in Fig. 3, where the conversion process includes three steps, namely,

(1)for each vertex $v_j \in V$, create a pair of vertices LT_j, UT_j, and a pair of dashed edges (LT_j, UT_j; *LT to UT*) and (UT_j, LT_j; *UT to LT*),

(2)for each edge $(T_m; v_i, v_k; i_p/o_q) \in E$, create fines edges as follows:

(UT_j, UT_k; i_p/o_q), if both i_p and o_q are related to UT, or
(UT_j, LT_k; i_p/o_q), if i_p is related to UT and o_q is related to LT, or
(LT_j, LT_k; i_p/o_q), if both i_p and o_q are related to LT, or
(LT_j, UT_k; i_p/o_q), if i_p is related to LT and o_q is related to UT, and

(3)remove all isolated vertices.

In step (1), each vertex LT_j (or UT_j) stands for both the starting place of a test segment with first input operation related to LT (or UT) and the starting state being v_j, and each dashed edge (LT_j, UT_j) (or (UT_j, LT_j)) represents an external synchronization operation *LT to UT* (*UT to LT*). In step (2), each fine edge represents a transition of M. In step (3), the vertices that are not accessed by any path are removed. Thus, the shortest paths from UT_j (or LT_j) to UT_j (or LT_j) can be used to generate the minimum-cost (or-length) synchronizable transfer sequences which transfer M from state v_j to state v_k.

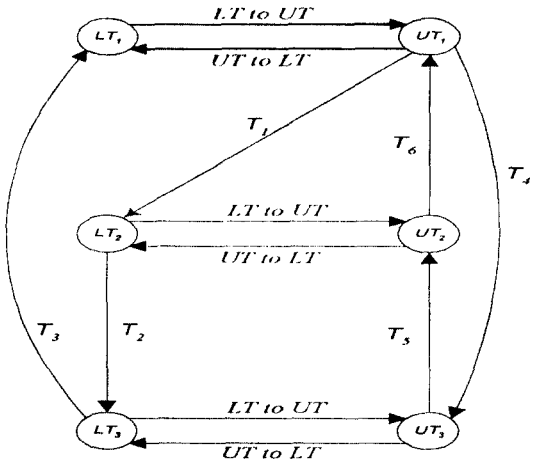

Fig. 3. The Duplex Digraph G' (V', E') Converted From the Graph $G(V, E)$ of M in Fig. 2.

Ⅴ. Minimum-Length Synchronizable Test Sequence Generation

In this paper, it is assumed that M is a deterministic and minimal FSM M which is represented by strongly connected graph $G(V, E)$. The proposed method for generating a minimum-cost (or-length) synchronizable test sequence for a given FSM M will be constructed as the following four phases:

(1)The graph $G(V, E)$ of an FSM M is converted into a duplex digraph G' (V', E'). The conversion process for each vertex and edge in the graph $G(V, E)$ is as shown in Section 2.

(2)A set of LT- and UT- synchronizable distinguishing sequences for each state of M is constructed. An *LT-(UT-) synchronizable DS* (v_j), denoted as *LT-(UT-)DS* (v_j), is a distinguishing sequence which starts at state v_j with an input operation related to LT (UT) or an external synchronization operation *LT to UT* (*UT to LT*). A set of LT- and UT-synchronizable distinguishing sequences for each state of the FSM in Fig. 2 is shown in Table 3. The *LT-(UT-)DS* (v_j) sequences can be obtained from an

input@DS-Diagram [10, 11] denoted by $G[E_c]=(V, E_c)$. When a distinguishing sequence $[a, a]$ is used, the diagram $G[E_c]$ of an FSM M is shown in Fig. 4 and the test segments with the form *input@DS* are listed in Table 2.

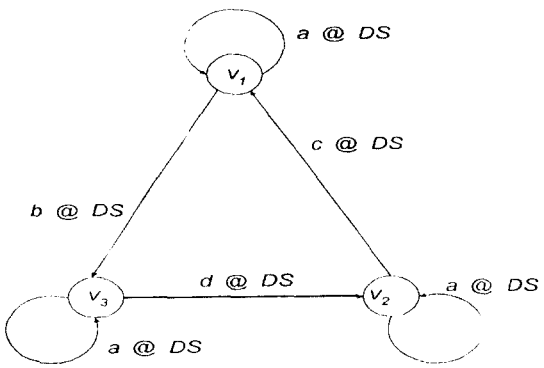

Fig. 4. *Input @DS-Diagram* $G[E_c]$ of M in Fig. 2.

Table 2. Test segment [*input @DS*] for M in Fig .2

$DS=a, a$

T_1: (1, 2; *b/y*) (2, 1; *a/z*) (1, 3; *a/x*) = (1, 3; *b/y @DS*(v_2))
T_2: (2, 3; *c/y*) (3, 2; *a/z*) (2, 1; *a/z*) = (2, 1; *c/y @DS*(v_3))
T_3: (3, 1; *d/y*) (1, 3; *a/x*) (3, 2; *a/z*) = (3, 2; *d/y @DS*(v_1))
T_4: (1, 3; *a/x*) (3, 2; *a/z*) (2, 1; *a/z*) = (1, 1; *a/x @DS*(v_3))
T_5: (3, 2; *a/z*) (2, 1; *a/z*) (1, 3; *a/x*) = (3, 3; *a/z @DS*(v_2))
T_6: (2, 1; *a/z*) (1, 3; *a/x*) (3, 2; *a/z*) = (2, 2; *a/z @DS*(v_1))

Table 3. A Set of Synchronizable Distinguishing Sequences (*LT-*(*UT-*)*DS*(v_j)) for Each State of M in Fig. 2

v_1: *UT-DS*(v_1) = (UT_1^*, UT_2; *a/x, a/z*)
LT-DS(v_1) = (LT_1^*, UT_2; *LT to UT, a/x, a/z*)
v_2: *UT-DS*(v_2) = (UT_2^*, UT_3; *a/z, a/x*)
LT-DS(v_2) = (LUT_2^*, UT_3; *LT to UT, a/z, a/x*)
v_3: *UT-DS*(v_3) = (UT_3^*, UT_1; *a/z, a/z*)
LT-DS(v_3) = (LT_3^*, UT_1; *LT to UT, a/z, a/z*)

(3) A duplexD digraph $G\hat{}(V\hat{}, E\hat{})$, as shown in Fig. 5, for the FSM M is created by adding a set of bold edges and vertices to the duplex digraph $G'(V', E')$ according to the following steps, namely,

(a) for each edge $(T_m; v_j, v_k; i/o) \in E$, add a bold vertex Y_k^* and a bold edge $(T_m; X_j, Y_k^*; i/o)$, where
X = *LT* if i is related to LT, X = *UT* if i is related to UT, and
Y = *LT* if both i and o are related to LT, or
Y = *UT* if both i and o are related to UT, or
Y = *LUT* if i is related to LT (or UT) and o is related to UT (or LT),

(b) for each Y_k^* and each Y-*DS*(v_k) = (v_k, v_h; *i/o-sequence*), add
a bold-dashed edge (Y_k^*, LT_h; *i/o-sequence*) if [*i/o-sequence, LT-related* i_p] is synchronizable, or
a bold-dashed edge (Y_k^*, UT_h; *i/o-sequence*) if [*i/o-sequence, UT-related* i_p] is synchronizable, and

(c) for each LUT_k^* and each *LT-DS*(v_k) (or *UT-DS*(v_k)) = (v_k, v_k; *i/o-sequence*), add
a bold-dashed edge (LUT_k^*, LT_h; *i/o-sequence*) if [*i/o-sequence, LT-related* i_p] is synchronizable, or
a bold-dashed edge (LUT_k^*, UT_h; *i/o-sequence*) if [*i/o-sequence, UT-related* i_p] is synchronizable.

In step (3) of the above procedure, each bold edge $(T_m; X_j, Y_k^*; i/o)$ represents the first i/o pair in the test segment to verify transition T_m. Each bold vertex LT_k^* (or UT_k^*) is where a bold edge with operations *LT-related* i_p/*LT-related* o_q (*UT-related* i_p/*UT-related* o_q) ends, and where an LT-(UT-) synchronizable distinguishing sequence starts. Each bold vertex LUT_k^* is where a bold edge with operations related to both testers ends, and where either an operation LT- or UT- synchronizable distinguishing sequence starts. Each bold-dashed edge (Y_k^*, v_k) represents a synchronizable *DS*(v_k) of state v_k.

(4) The RCPT algorithm [6] is employed to construct an RCPT in the duplexD digraph $G\hat{}(V\hat{}, E\hat{})$ over the set of edges representing the test segments. As a result, a minimum-cost (or-length) tour that traverses every bold edge at least once of the duplexD digraph $G\hat{}(V\hat{}, E\hat{})$ covering all the nonoverlapped test

segments becomes a minimum-cost (or -length) synchronizable test sequence for the FSM M. For an example as shown in Fig. 5, the RCPT over the bold edges is as follows: [T_4, *UT-DS*(v_3), T_1, *LT-DS*(v_2), T_5, *UT-DS*(v_2), *UT to LT*, T_3, *LT-DS*(v_1) T_6, *UT-DS*(v_1), T_6, T_1, T_2, *LT-DS*(v_3)]. This tour is used to generate the test sequence [*a/x, a/z, a/z, b/y, LT to UT, a/z, a/x, a/z, a/z, a/x, UT to LT, d/y, LT to UT, a/x, a/z, a/z, a/x, a/z, a/z, b/y, c/y, LT to UT, a/z, a/z*] with the total cost of 20 input/output operations and 4 external synchronization operations, where the test segment for T_4 is [*a/x, a/z, a/z*], for T_1 is [*b/y, LT to UT, a/z, a/x*], for T_5 is [*a/z, a/z, a/x*], for T_3 is [*d/y, LT to UT, a/x, a/z*], for T_6 is [*a/z, a/x, a/z*], and for T_2 is [*c/y, LT to UT, a/z, a/z*]. Normally, it is assumed that each input/output operation (i_p/o_q) takes a unit cost, each external synchronization operation (*LT to UT* or *UT to LT*) takes 5 units cost, and each manual coordination by use of a telephone or terminal connection takes 10 units cost. In case of the above example, the total cost of the test sequence with external synchronization operations is 40 units. On the other hand, the total cost of the test sequence with manual coordination is 60 units. Therefore, the former takes about 30% saving in the unit cost, compared with the latter.

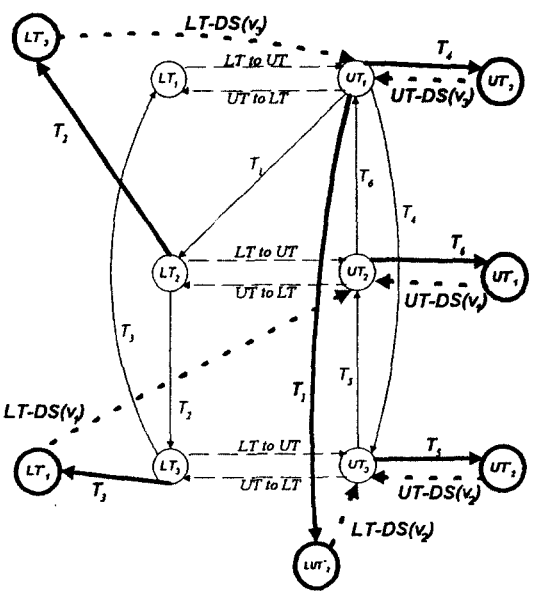

Fig. 5. The DuplexD Graph $G\hat{}(V\hat{}, E\hat{})$ Constructed From the Duplex Graph G' (V', E') in Fig. 3.

Ⅵ. Complexity of the Proposed Method

In the proposed method, a synchronizable test sequence is generated from $G(V, E)$ as follows: (1) The digraph G is converted into a duplex digraph G' (V', E'). (2) Test segments with the form *input @DS* are constructed from the $G[E_c]$ diagram. Synchronizable distinguishing sequences (*LT-* (*UT-*) $DS(v_j)$) are obtained from the $G[E_c]$ diagram and these test segments. (3) A duplexD digraph $G\hat{}(V\hat{}, E\hat{})$ is created by adding edges (representing these test segments and synchronizable distinguishing sequences) to G'. (4) The RCPT algorithm is employed to construct an RCPT in the duplexD digraph $G\hat{}(V\hat{}, E\hat{})$ over the set of edges representing the test segments. Then, the RCPT is a minimum-cost (or -length) synchronizable test sequence for the given FSM M. Assume that n and m are the number of vertices and edges in G, respectively, n', m' and m_e' are the number of vertices, edges and edges for external synchronization operations, respectively, and $n\hat{}$, $m\hat{}$ and $m_e\hat{}$ are the number of vertices, edges and bold edges in $G\hat{}$, respectively. Then $O(n\hat{}) = O(2n\hat{}) = O(4n)$, $O(m\hat{}) = O(m') = O(m)$, and $O(m_e\hat{}) = O(2m_e') = O(4n)$. Step (1) requires $O(n' + m' + m_e') = O(4n + m)$. According to [10, 11], step (2) takes $O((n-1)n^n)$. Step (3) requires $O(n\hat{} + m\hat{} + m_e\hat{}) = O(8n + m)$. Step (4) can be reduced to the minimum-cost maximum flow problem which then takes $O(n\hat{}(m\hat{} + n\hat{}\log n\hat{})\log(m_e\hat{})) = O(4n(m + 4n\log 4n)\log(4n/m))$ [6]. The total complexity is $\min(O((n-1)n^n), O(n(m + n\log n)\log(n/m)))$.

Theorem 1: If a graph representaiton $G(V, E)$ of IUT has a property of input/output interactions related to LT and/or UT, a tour of the duplex digraph

$G'(V', E')$ converted from G yields synchronizable sequence.

Proof: By definition 3, it is obvious that a test sequence generated form a tour of this duplex digraph will be $[i_1/o_1, X_1, i_2/o_2, X_2, ..., X_{n-1}, i_n/o_n]$ with both operations (o_k and X_k) and (X_k and i_{k+1}) being related to LT (or UT) and UT (or LT), respectively. Therefore, any of the duplex digraph $G'(V', E')$ can be employed to generate a synchronizable test sequence.

Theorme 2: If we construct an Euler tour P of a rural symmetric augmentation G^* in the duplexD digraph $G\hat{}(V\hat{}, E\hat{})$, the tour P corresponds to a rural Chinese postman tour (RCPT) which is a minimum-cost (or -length) synchronizable test sequence for the given FSM M.

Proof: Assume on the contrary that there exists an RCPT T with cost $C(T) < C(P)$. Let $X\hat{}(v_j, v_k:i_p/o_q)$ by the number of times $(v_j, v_k:i_p/o_q)$ is traversed in T and $X(v_j, v_k:i_p/o_q)$ by the number of times $(v_j, v_k:i_p/o_q)$ is traversed in P. A directed graph G^* is constructed by including $X\hat{}(v_j, v_k:i_p/o_q)$ copies of edge $(v_j, v_k:i_p/o_q) \in E\hat{}$. Since each edge in E_c is traversed at least once and the starting and ending vertex is v_1, the resulting graph G^* is symmetric and contains each edge of E_c at least once. Since $C(X\hat{}) = C(T) < C(P) = C(X)$, X is not rural symmetric augmentation, which is contradiction.

Ⅶ. Conclusion

In this paper, the problem has been studied for generating a synchronizable test sequence that can be applied in the distributed test architecture for testing a protocol implementation, presented a new technique for generating a minimum-length synchronizable test sequence that can be applied in a testing system where the cost of both external synchronization operations and input/output operations are taken into consideration. The method defines a set of transformation rules that constructs a duplex digraph from a given finite state machine representation of a protocol specification such that a rural Chinese postman tour of the duplex digraph can be used to generate a minimum-length synchronizable test sequence using synchronizable distinguishing sequences as the state identification sequence for each state of the given finite state machine.

Several issues of research in synchronizable test generation remain open. In this paper, it is assumed that the method above is based on the deterministic finite state machine model. We have not considered non-determinism problem as most specifications written in formal description techniques normally have. Synchronizable test generation from non-deterministic finite state machines is still a research topic. Also, more research covering both the control and data flow aspects of protocols is required. Both problems are important in the protocol conformance testing.

References

1. D. Rayner, "OSI Conformance Testing," *Computer Networks and ISDN Systems*, vol. 14, no. 1, pp. 79-98, 1987.
2. W.H. Chen et al., "Synchronizable Protocol Test Generation via the Duplex Technique," in: *Proc. IEEE INFOCOM '90*, pp. 561-563, 1990.
3. B. Sarikaya and G. v. Bochmann, "Synchronization and Specification Issues in Protocol Testing," *IEEE Transactions on Communications*, vol. 32, no. 4, pp. 389-395, 1984.
4. H. Ural and Z. Wang, "Synchronizable Test Sequence Generation Using UIO Sequences," *Computer Communications*, vol. 16, no. 10, pp. 653-661, 1993.
5. H. Motteler et al., "Undetected Faults in Protocol Testing," *IEEE Transactions on Communications*, vol. 43, no. 8, pp. 2289-2297, 1995.
6. A.V. Aho et al., "An Optimization Technique for Protocol Conformance Test Generation Based on UIO Sequences and Rural Chinese Postman Tours,"

IEEE Transactions on Communications, vol. 39, no. 11, pp. 1604-1615, 1991.

7. A.T. Dahbura et al., "Formal Methods for Generating Protocol Conformance Test Sequences," in: Proc. The IEEE, vol. 78, no. 8, pp. 1317-1326, 1990.
8. D.P. Sidhu and T.K. Leung, "Formal Methods for Protocol Testing: A Detailed Study," *IEEE Transactions on Software Engineering*, vol. 15, no. 4, pp. 413-426, 1989.
9. ISO, Information Processing Systems-*OSI Conformance Testing Methodology and Framework*, IS 9646, Parts 1-5, 1991.
10. G. Gonenc, "A Method for the Design of Fault Detection Experiments," *IEEE Transactions on Computers*, vol. C-19, pp. 551-558, 1970.
11. Z. Kohavi, Switching and *Finite Automata Theory*, McGraw-Hill, New York, 1978.

Chul Kim 정회원

Chul Kim received the B. S. and M. S. degrees in electronics engineering from Yonsei University and Kyunghee University, Korea, in 1977 and 1979, respectively.

From 1981 to 1983, he was a Research Engineer at the Samsung Telecommunications Research Institute, Korea. From 1984 to 1993, he was with IBM Korea Inc. as Advisory Systems Engineer of communication networks and operating systems, and Program Manager of Information Technology (IT) standards. He is presently an Assistant Professor in the Department of Computer Science and Statistics, Yongin University, Korea. Since 1990, he has also served on the standardization committees of ISO/JTC1 and ITU-TS in Korea. His research interests include computer networks, protocol engineering, and coded character sets for information processing systems.

Joo Seok Song 정회원

Joo Seok Song received the B. S. degree in electrical engineering from Seoul National University, Korea, in 1976, the M. S. degree in electrical and electronics engineering from Korea Advanced Institute of Science and Technology, in 1979, and the Ph. D. degree in computer science from University of California at Berkeley in the United States, in 1988.

From 1979 to 1982, he was a researcher at the Electronics and Telecommunications Research Institute in Korea. In 1988, he was an Assistant Professor of Naval Postgraduate School in the United States. He is currently a Full Professor in the Department of Computer Science, Yonsei University, Korea. Since 1990, he has also served on the consultative committee of National Financial Computer Network and the standardization committee of ITU-TS in Korea. His current research interests include high speed network, conformance testing, computer security, and distributed computing systems.