14

A Generalization of the multiple UIO method of test sequence selection for protocols represented in FSM

T. Ramalingam[a] K.Thulasiraman[a] and Anindya Das[b]

[a]Electrical Engineering Department, Concordia University, Montreal, Canada
[b]D.I.R.O., University of Montreal, Montreal, Canada

We present a generalized method based on the MU-method [30, 29] for selecting test sequence for conformance testing of protocol implementations whose specifications are represented as deterministic Finite State Machines (FSM). Our method uses solutions to two subproblems: Basic UIO Assignment Problem (BUAP), and the Rural Postperson Problem (RPP). The BUAP is defined and an efficient algorithm based on the algorithm for the maximal cardinality two-matroid intersection problem [20] is presented. A heuristic algorithm is proposed for the general RPP, which is known to be NP-complete. The method works for all strongly connected FSM-based protocols which have at least one UIO-sequence for each state whereas the U-method [4] and the MU-method are applicable only for a subset of this class of protocols. The proposed method produces test sequences of varied lengths depending on the structure of the protocols as well as the set of UIO-sequences used in selecting the test sequence.

1 Introduction

Conformance Testing is intended to assure that a given implementation of a protocol is equivalent to the standard specification of the protocol [32, 2]. The goal of interworking among heterogeneous systems can be achieved through conformance testing. Test suite selection (generation) is an important problem as the efficiency and the quality of testing depends on the test suite selected. The *OSI conformance testing methodology and framework* [2] defines a test suite as a set of test cases, one for each test purpose. Each test case is a collection of event sequences. If a protocol is specified as a deterministic Finite State Machine (FSM) then its test suite is usually specified as a single sequence of labels (input and expected output pairs) of the transitions. Such a test suite is referred to as a *test sequence* [4]. The specification of a protocol is normally described in Formal Description Techniques (FDTs) such as LOTOS [8], Estelle [9], or SDL [1]. Such a specification has many advantages including automatic test suite selection [3]. The control structure in the specification of a protocol can be represented as a Finite State Machine [6, 21]. In this paper, we consider only the control flow aspect of testing and we assume that the protocol is represented as an FSM.

Among the various FSM-based test sequence selection methods [4, 31, 10, 17, 34, 35], the U-method by Aho *et al* [4] is well known for its minimal length test sequence with high fault coverage [24]. The method uses the criterion of covering each transition and confirming its tail state. UIO-sequences (defined later) are used for confirming the tail

T. Mizuno et al. (eds.), *Protocol Test Systems*

states of transitions. This method generates a minimal length test sequence satisfying the coverage criterion by formulating and solving a special type of *Rural Postperson Problem* (RPP) [15] for which an efficient solution exists [4]. Further minimization of the length of the test sequence is achieved in the MU-method [30] by assigning a suitable UIO-sequence for each transition from a set of UIO-sequences for each state. As pointed out in Section 3, these methods are, however, not applicable for all protocols having UIO-sequences for all states. In this paper we generalize the MU-method so that it can be applied to select test sequences for all such protocols. The protocols considered in this paper are represented as strongly connected FSMs having at least one UIO-sequence for each state. The generalized method requires solutions to two sub-problems: Basic UIO Assignment Problem (BUAP), and the general RPP. The BUAP is defined and an efficient solution based on the maximum cardinality two-matroid intersection algorithms by Lawler [19, 20] and Edmonds [14] is presented. A heuristic algorithm is proposed for the RPP; an upper bound on the optimality of the tour is also established for this algorithm. Our method carefully combines the above two algorithms with the MU-method to derive test sequences for the given protocol. The method produces test sequences of varied lengths depending on the structure of the protocols as well as the set of UIO-sequences considered.

The required definitions and the UIO-based test sequence selection methods are introduced in Section 2. Motivations for our research are presented in Section 3. In Section 4 we define the BUAP and provide an efficient algorithm. The heuristic algorithm for the RPP is developed in Section 5. Section 6 presents our generalized UIO testing method. The results are summarized in Section 7.

2 Preliminaries

2.1 Graphs and Matroids

Given a strongly connected directed graph $G = (V, E)$ with weighted edges, and a subset of edges $F \subseteq E$, the *asymmetric Rural Postperson Problem* (RPP) with respect to F is to find a tour with minimum cost such that it covers each edge in F at least once [15, 33]. Such a minimum cost tour is referred to as a *Rural Postperson Tour* (RPT) with respect to F. The RPP is known to be an NP-complete problem [25].

The subgraph of $G = (V, E)$ induced by an edge set $F \subseteq E$ is denoted by $G[F]$. If K is a set of edges having both end vertices in V, then $G + K$ denotes the graph obtained from $G = (V, E)$ by adding all the edges in K to G.

A *matroid* $M = (E, \mathcal{I})$ is a structure in which E is a finite set of elements and $\mathcal{I}$ is a family of subsets of E such that (i) empty set is a member of $\mathcal{I}$; (ii) if $F_1 \subset F$ and $F \in \mathcal{I}$ then $F_1 \in \mathcal{I}$; and (iii) if F_p and F_{p+1} are sets in $\mathcal{I}$ having p and $p+1$ elements respectively, then there exists an element $e \in F_{p+1} - F_p$ such that $F_p \cup \{e\} \in \mathcal{I}$. Each element in $\mathcal{I}$ is called an independent set in M. Let $M_1 = (E, \mathcal{I}_1)$ and $M_2 = (E, \mathcal{I}_2)$ be two matroids. The *Maximum Cardinality Two Matroid Intersection Problem (MC2MIP)* [20] is to find a set $H \subseteq E$ of maximum cardinality such that H is independent in both M_1 and M_2.

A *graphic matroid* of a graph $G = (V, E)$ is a matroid $(E, \mathcal{I})$ such that $F \subseteq E$ is in $\mathcal{I}$ iff F contains no cycle in G.

Let $P = \{E_1, E_2, \ldots, E_k\}$ be a partition of the edge set E of a graph $G = (V, E)$. Let $Q = \{i_1, i_2, \ldots, i_k\}$ be a given set of non-negative integers. Let $(E, \mathcal{I})$ be a system such that $F \in \mathcal{I}$ iff $|E_j \cap F| \leq i_j$ for $j = 1, 2, \ldots, k$. It is known that $(E, \mathcal{I})$ is a matroid [20]. It is called a *partition matroid* of G with respect to the partition P and the index Q.

2.2 Finite State Machines

An FSM M can be formally defined as a 5-tuple $M = (S, s_1, I, O, T)$ where S is the nonempty set of states of M in which s_1 is a designated state called the *initial state*. I and O are nonempty sets of possible inputs and outputs of the protocol, respectively. The transition function T is a partial function defined as $T : S \times I \to S \times O$. $T(s_i, a) = (s_j, o)$ means that the FSM M at state s_i makes a transition to state s_j when the input a is applied producing the output o. Graphically this is also represented as $s_i - a/o \to s_j$.

An FSM $M = (S, s_1, I, O, T)$ can also be represented by a directed labeled graph $G_s = (V, E)$, where $S = V$ and each transition $s_i - a/o \to s_j$ corresponds to an edge in E directed from s_i to s_j with label a/o. Thus an edge in E is specified by a triple $(s_i, s_j; a/o)$. We assume that the functions $start(e)$, $label(e)$ and $end(e)$ will return the starting state, label and the ending state of any edge e, respectively. We assign a unit cost with each transition since we focus on the test sequence length minimization. An FSM is said to have *reset capability* if for each state s_i in S there exists a transition $(s_i, s_1; r/-)$, called a *reset transition* which resets the FSM to its initial state where 'r' denotes the 'reset' command and '$-$' denotes that the FSM does not produce any output for the reset command.

A sequence of input-output is a concatenation of input-output pairs. We use the operators $\bullet$ and @ for concatenating input-output symbols and input-output sequences, respectively. These operators are omitted in certain sequences whenever there is no confusion. A sequence q of input-output pairs is said to be *applicable* at a state s_i of an FSM if the output part of q is observed on applying the input part of q to the FSM at the state s_i.

More formally, a sequence $q = a_1/o_1 \bullet a_2/o_2 \bullet \ldots \bullet a_l/o_l$ is applicable at state s_i iff $\exists s_{i_j}, j = 1, 2, \ldots l, l \geq 1$ such that $s_i - a_1/o_1 \to s_{i_1}$ and $s_{i_{j-1}} - a_j/o_j \to s_{i_j}$ for $2 \leq j \leq l$.

An *Unique Input Output (UIO) sequence* for a state s_i is an input-output sequence of minimum length such that it is applicable only at s_i of G_s. Note that for each UIO-sequence of the state s_i, there is a unique path from s_i. For better understanding, the UIO-sequences in the examples are expressed as the concatenation of the transitions along their corresponding paths. We assume that functions *head*, *tail*, and *length* will return the starting state, ending state, and the number of input-output pairs of any UIO-sequence, respectively.

Let MU_i be a nonempty set of UIO-sequences for each state s_i of the specification graph $G_s = (S, E)$. Let $MU = MU_1 \cup MU_2 \cup \ldots \cup MU_n$. Define the relation $R \subseteq E \times MU$ such that $(e, u) \in R$ iff $end(e) = head(u)$. Clearly, R denotes the set of all possible assignments of UIO-sequence from MU for all the transitions in E. We call any subset $B \subseteq R$ a *valid UIO assignment* or simply an *UIO assignment* for the set of transitions $D \subseteq E$ if $dom(B) = D$ and $|\{u|(e, u) \in B\}| = 1$, for each $e \in D$. That is, each element in D has exactly one UIO-sequence assigned in B. A valid UIO assignment for E is also referred to as a *(valid) UIO assignment of the protocol* G_s. Consider the undirected graph $G' = (S, E')$ where $E' = \{(start(e), tail(u); label(e)@u)|(e, u) \in R\}$. It is easy to see that there is a one-to-one correspondence between R and E'. In this paper, an element of R is often treated as an edge in E' and vice versa. An edge in E' is often referred to as a *test edge* of the underlying transition. For each edge $e' = (start(e), tail(u); label(e)@u) \in E'$ which corresponds to $(e, u) \in R$, the length of the input sequence in $label(e)@u$ is taken as the cost of e'. Let B be a valid UIO assignment for $D \subseteq E$. The subgraph $G'[B]$ of G' induced by B is called a *test graph* for D. The test graph induced by an UIO assignment of the protocol G_s is simply referred to as a *test graph for the protocol.* Observe that every test graph of a strongly connected protocol always spans all the states of the protocol. In this paper, subgraphs of G' are often extended by adding edges from G_s, and vice versa. Suppose H' is a subgraph of G' and $F \subseteq E$, then $H' + F$ will be treated as an undirected graph as H' is undirected. On the other hand, if H is a subgraph of G_s and $F' \subseteq E'$, then $H + F'$ will be treated as a directed graph as H is directed, the orientation of the edges in F' coinciding with that of the corresponding edge in G_s.

2.3 Testing Methods

The U-method [4] introduced in [28] requires that the representation graph $G_s = (S, E)$ be strongly connected. Each state of G_s is assumed to have an UIO-sequence. Let U_j be an UIO-sequence for s_j, $1 \leq j \leq n$. The U-method tests each transition $(s_i, s_j; a/o)$, as follows

> The protocol *Implementation Under Test* (IUT) is first put in state s_i . Then the input a is applied and the output is verified for o. Finally, to check for state s_j the input part of U_j is applied to the current state of the IUT and the output is examined against the output part of U_j

The input-output sequence $a/o@U_j$ is the test subsequence for the transition $(s_i, s_j; a/o)$. By considering $MU_j = \{U_j\}$, $1 \leq j \leq n$, we get $G' = (S, E')$, where $E' = \{(s_i, tail(U_j); a/o @U_j) \mid (s_i, s_j; a/o) \in E\}$. Clearly, G' is the unique test graph of G_s. Let $G^* = G_s + E'$. In the U-method, each transition in G_s is tested by applying the subsequence along its test edge in E'. Thus the optimal test sequence for G_s lies along the RPT of G^* with respect to E'. In other words, the optimal test sequence selection problem is equivalent to the problem of finding an RPT of G^* with respect to E'. Before proceeding further, we introduce a definition. A *rural symmetric augmentation* of a weighted graph $G = (V, E)$ with respect to $F \subseteq E$ is a graph $G[F \cup E_1]$ such that (i) number of incoming edges at each vertex in $G[F \cup E_1]$ is the same as the number of outgoing edges from that state, and (ii) E_1 is a *bag*[1] of minimum cost in E satisfying (i). The polynomial algorithm given in [28] for finding an RPT first computes a rural symmetric augmentation $G^*[E' \cup E_1]$ of G^* with respect to E', where E_1 is a bag in E. It then generates a test sequence by concatenating the subsequences and/or labels along an euler tour of $G^*[E' \cup E_1]$. This algorithm can be successfully applied to a protocol G_s if the test graph G' is connected [4]. Note that this is only a sufficient condition. It is also shown that protocols which have either a self-loop at each state or the reset capability always meet this requirement.

In the MU-method [30, 29], Shen *et al* have recently proposed an improvement for the U-method. While the U-method uses only one UIO-sequence for each state, this method uses multiple ($\geq$ 1) UIO-sequence(s) for each state. The improvement is obtained by suitably assigning an UIO-sequence for each transition from the set of multiple UIO-sequences of its tail state. Given a set MU_i of multiple UIO-sequences of minimal length for each state $s_i, i = 1, 2, \ldots n$ of the protocol $G_s = (S, E)$, the *UIO-sequence Assignment Problem (UAP)* is to find a valid UIO assignment B of the protocol such that the RPT of $G_s + B$ with respect to B is of minimum length among all valid UIO assignments of the protocol. The MU-method solves certain specific instances of this problem efficiently by transforming it into an equivalent multi-stage minimum cost maximum flow problem [29]. As in the U-method, a minimum length test sequence is obtained by concatenating the test subsequences and/or input-output of transitions along the minimum cost RPT. The MU-method guarantees an optimal test sequence for a protocol G_s if the test graph

[1] A bag is a collection of elements over some domain. Unlike sets, bags can have multiple occurrences of the same element

$G'[B]$ is connected. It has been proved that the protocols which have either the reset capability or a self-loop at each state always meet this requirement [29]. As we shall see in Section 3, this approach of obtaining minimum length test sequences does not work for all protocols.

Methods for further minimizing the length of a test sequence by overlapping test subsequences of the transitions are presented in [11, 23, 22]. Our focus in this paper is on the applicability of the UIO-based approaches on different protocols and we study the U-method and the MU-method for this purpose. Henceforth, the U-method and the MU-method are referred to as the UIO-based methods. We need some more definitions.

Assume that the implementations may have only two types of faults: *output fault, transfer fault* [5]. Informally, an implementation is said to have an output fault (transfer fault) in a transition, if the transition produces an output (terminates at a state) different from the expected one as per the specification. The *fault coverage* of a test sequence is the ratio of the number of faulty implementations the test sequence can detect to the total number of possible faulty implementations. A test sequence selection method is said to have *complete fault coverage* if the fault coverage of any test sequence selected by this method is 1.

3 Motivation for the Present Work

It has been reported in [4] and [13] that the U-method can be applied to select test sequence for any protocol G_s which satisfies one of the conditions (i) through (v) below. Note that conditions (i) through (iv) are independent of UIO-sequences whereas condition (v) is with respect to a particular UIO-sequence for each state.

(i) G_s has the reset capability [4].

(ii) G_s has a self-loop at each state [4].

(iii) G_s has a state, say s_e, with a self-loop and a reset edge, and each state has a self-loop, or a reset edge, or an edge to the state s_e [13].

(iv) For every partition of S into two nonempty subsets S_A and $S - S_A, \exists s_i \in S_A$ and $s_j \in S - S_A$ such that there is an edge to some state s_k from both s_i and s_j [13].

(v) For every partition of S into two nonempty subsets S_A and $S - S_A, \exists s_i \in S_A$ and $s_j \in S - S_A$ such that state $s_i(s_j)$ has an edge to a state $s_p(s_q)$ in S and $tail(U_p) = tail(U_q)$. Here, U_j is an UIO-sequence for the state s_j, $j = 1, 2, \ldots, n$ and it is used for testing every incoming transition at the state s_j [13].

We would like to note that there are real life protocols, for example a simplified transport protocol as given in [7], which do not satisfy any of these conditions, yet the U-method can successfully select test sequences for these protocols provided suitable UIO-sequences are chosen [27].

Careful assignment of UIO-sequences to transitions is necessary since an arbitrary assignment may not produce a connected test graph despite the existence of such assignment. For example, consider the abstract FSM protocol as given in Figure 1, based on the *responder* module of the INRES protocol [18]. Only the core transitions are considered here. The states s_1, s_2, and s_3 correspond to the states *DISCONNECTED, WAIT*, and *CONNECTED* of the *responder* module, respectively. We have slightly modified the original labels of the transitions so that the FSM has multiple UIO-sequences. The labels of the

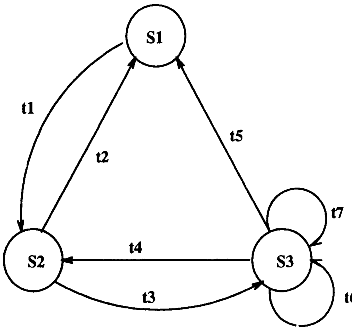


Figure 1: An FSM based on the INRES protocol: *responder*

Transition	Label	Transition	Label
t1	CR/ICONind1	t2	IDISreq/DR1
t3	ICONresp/CC	t4	CR/ICONind2
t5	IDISreq/DR2	t6	DT2/AK
t7	DT1/IDATind&AK		

Table 1: Labels of the transitions in Figure 1

transitions are given in Table 1. Let $MU_1 = \{t1\}$, $MU_2 = \{t2, t3\}$, and $MU_3 = \{t4, t5, t6\}$ be the set of UIO-sequences for the states s_1, s_2, and s_3, respectively. Note that the UIO-sequences are denoted by their corresponding transitions. Let $MU = MU_1 \cup MU_2 \cup MU_3$. Clearly, the assignments A_1 and A_2 given in Table 2 and Table 3 , respectively, are valid UIO assignments of the protocol. Also these assignments are solutions to the UAP. Note that the test graph $G'[A_1]$ is not connected whereas the other test graph $G'[A_2]$ is connected. If the UIO-based methods assign UIO-sequences to the transitions as per A_1 then they cannot select a test sequence for this protocol. On the other hand, A_2 facilitates the UIO-based methods to select an optimal test sequence for the protocol. The above discussion implies that the UIO-based methods may not always produce a test sequence even if the protocol has a connected test graph. Unfortunately, there is no way to ensure that the min-cost max-flow approach will lead to a graph which is connected.

It should also be emphasized that certain protocols may not even have any connected test graph. Consider the FSM representation of a simplified alternating bit protocol (*receiver*) shown in Figure 2. $m0/a0$ and $m1/a1$ are the only UIO-sequences for the states s_1 and s_2, respectively. The FSM neither satisfies the requirement stated in conditions (i) through (v) nor has a valid UIO assignment so that the resulting test graph is connected.

Thus the following questions arise: Given a set of multiple UIO-sequences for each state, does the protocol have a set $BE \subseteq E$ of transitions and a valid UIO assignment for BE such that the resulting test graph for BE is connected one which spans all the

Transition	UIO-sequence	Transition	UIO-sequence
t1	t2	t2, t5	t1
t3, t6, t7	t6	t4	t3

Table 2: Valid UIO assignment without connected test graph

Transition	UIO-sequence	Transition	UIO-sequence
t1	t3	t2, t5	t1
t3, t6, t7	t6	t4	t2

Table 3: Valid UIO assignment with connected test graph

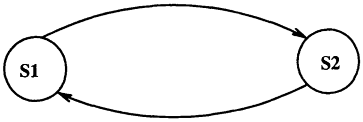


Figure 2: Simplified alternating bit protocol (*receiver*)

states of the protocol? If so, how to find a minimal set of transitions satisfying the above condition? (This problem is formalized in Section 4 as the Basic UIO Assignment Problem (BUAP).) If not, how to generate a test sequence for this protocol. These questions are addressed in Section 5 and Section 6. While the BUAP assigns UIO-sequences for a subset of the transitions in G_s, the min-cost max-flow problem formulated in Section 6 assigns UIO-sequences for the remaining transitions such that the overall length of the test sequence is minimized.

In this paper, we propose a new method which can be used to select test sequence from any FSM-based strongly connected protocol having at least one UIO-sequence for each state; the proposed method applies optimization techniques at various phases in order to minimize the length of the test sequence. Our method is a generalization of the MU-method and it is based on the BUAP and the general RPP. We present an efficient algorithm for the BUAP and a heuristic algorithm for the RPP before describing our test selection method.

4 Basic UIO Assignment Problem

As defined earlier, let MU_i be a nonempty set of UIO-sequences for each state s_i of the strongly connected specification graph $G_s = (S, E)$; $MU = MU_1 \cup MU_2 \cup \ldots \cup MU_n$. $R \subseteq E \times MU$ is a relation such that $(e, u) \in R$ iff $end(e) = head(u)$. Consider the undirected graph $G' = (S, E')$ where $E' = \{(start(e), tail(u); label(e)@u) | (e, u) \in R\}$. Observe that for each valid UIO assignment B, the induced graph $G'[B]$ is a test graph for $dom(B)$.

> **The Basic UIO Assignment Problem (BUAP) is to find a minimum set $K \subseteq E$ and a valid UIO assignment B of K such that $G'[B]$ has the minimum number of connected components spanning G'.**

The BUAP can be efficiently solved using the matroid theoretic approach. We demonstrate this by mapping the BUAP into an equivalent maximal cardinality two-matroid intersection problem which is solvable in polynomial steps. To start with, let us assume that G' is connected and it has no self-loop. Let $M_1 = (E', \mathcal{I}_1)$ be the graphic matroid of G'. Let Q_e be the set of all possible UIO-sequence assignments from MU for the transition e. Clearly, $Q_e \subseteq R$ and $dom(Q_e) = \{e\}$. Let $P = \{Q_e | e \in E\}$. Then clearly,

P is a partition of E'. Let $M_2 = (E', \mathcal{I}_2)$ be the partition matroid over the partition P and integers $i_e = 1$ for all $e \in E$. Suppose that I_{max} is a maximum set such that it is independent in M_1 as well as in M_2, then it is a valid assignment for $dom(I_{max})$ and $G'[I_{max}]$ is acyclic. Since I_{max} is a maximum set it spans G'. These properties in turn imply that $G'[I_{max}]$ contains the minimum number of components. As $G'[I_{max}]$ is acyclic no other valid UIO assignments of lesser cardinality will span G'. Hence, $dom(I_{max})$ and I_{max} form a solution to the BUAP.

We now present an efficient algorithm called *basic_assignment* for solving the BUAP. We assume that G' is connected and it has no self-loop. This algorithm is based on the algorithms [19, 14] for the maximal cardinality two-matroid intersection problem. While the algorithms given in [19, 14] are for any two matroids, we adapt their approach for the intersection of the graphic matroid M_1 and the partition matroid M_2 given above, thereby reducing the overall complexity of the algorithm. Our algorithm *basic_assignment* starts with an empty set of edges (that is, $H = \emptyset$). At each iteration of the **repeat...until** loop, the algorithm computes a valid UIO assignment H such that $G'[H]$ is acyclic and H has one element more than the number of elements it had in the previous iteration. The algorithm terminates when there is no such H in the current iteration. The UIO assignment H output by the algorithm and $dom(H)$ form a solution to the BUAP. A formal description of the algorithm is given below. This will be followed by an explanation for this algorithm. For the sake of simplicity in notation, we shall let an element $j = (e, u) \in E'$ also refer to the edge e.

```
Algorithm basic_assignment(G_s, MU, G', H);
{ Input: The digraph G_s = (S, E), graph G' = (S, E'), and a set of UIO-sequences MU }
{ Output: a set of edges H from E' }
   H ← ∅ ;
   V_H ← {s, t} ∪ E' ;
   repeat
      {Construct the digraph G_H = (V_H, E_H)}
      E_H ← ∅ ;
      for each j = (e, u) ∈ E' − H do
      begin
         if (G'[H ∪ {j}] is acyclic) then
            E_H ← E_H ∪ {(s, j)} ;
         if (e ∉ dom(H)) then
            E_H ← E_H ∪ {(j, t)} ;
         for each k = (e', u') ∈ H do
         begin
            if (e = e') then
               E_H ← E_H ∪ {(j, k)} ;
            if (G'[H ∪ {j}] has a unique cycle containing k) then
               E_H ← E_H ∪ {(k, j)} ;
         end
      end
      if (the digraph G_H = (V_H, E_H) has a path from s to t) then
      begin
         find a shortest path (s, j_1, k_1, ..., j_{p-1}, k_{p-1}, j_p, t) from s to t in G_H;
         H ← (H ∪ {j_1, j_2, ..., j_p}) − {k_1, k_2, ..., k_{p-1}}
      end
      else
      begin
         output(H);
         stop
      end
   for ever
end basic_assignment.
```

A particular iteration of the **repeat...until** loop first constructs a digraph $G_H = (V_H, E_H)$ for a given H. Here, $V_H = \{s, t\} \cup E'$, where s and t are two designated vertices in V_H. The set of vertices in V_H which represent the edges in E' is partioned into two sets: H and $E' - H$. The graph G_H is constructed in such a way that the presence of a path from s to t in this graph guarantees that the cardinality of H during the current iteration can be increased by one. In order to construct the edge set E_H, the following is done for each $j = (e, u) \in E' - H$.

> If $G'[H \cup \{j\}]$ is acyclic, then an edge from s to j is added to E_H. If $e \notin dom(H)$, then an edge from j to t is added to E_H. For each $k = (e', u') \in H$, an edge from j to k is added to E_H if k and j are test edges for the same transition (that is, if $e = e'$). Also, if j and k are contained in a cycle of $G'[H \cup \{j\}]$, then an edge is added to E_H from k to j.

If the digraph G_H has a path from s to t then let $(s, j_1, k_1, \ldots, j_{p-1}, k_{p-1}, j_p, t)$ be a shortest path from s to t. As established in Theorem 1, $H' = (H \cup \{j_1, j_2, \ldots, j_p\}) - \{k_1, k_2, \ldots, k_{p-1}\}$ is a valid assignment such that $G'[H']$ is acyclic. (Note that $|H'| = |H| + 1$.) Therefore, the algorithm proceeds to the next iteration of the **repeat...until** loop with H' as H. On the other hand, if G_H has no path from s to t, then the algorithm terminates since H computed in the previous iteration and $dom(H)$ form a solution to the BUAP (refer to Theorem 2).

Let n, m, and ν denote the number of states, the number of transitions and the maximum number of UIO-sequences in MU for any state of the protocol, respectively. Suppose that the computation needed to check if a given set is independent in a given matroid is considered as one step. Then, the algorithm *basic_assignment* requires $O(n(m\nu)^2)$ steps. Note that this complexity is better than the complexity $(O(m\nu)^3))$ of the general maximum cardinality two-matroid intersection algorithms. Our algorithm takes at most $O(n^2m^2\nu^2)$ time units when the time required to complete each step is also taken into account. In the presentation of the solution to the BUAP, we have assumed that G' is connected and it has no self-loop. The approach can easily be adapted for the general case [27]. The following theorems establish the correctness of the algorithm. Proofs for these theorems and detailed analysis of the algorithm are provided in [27].

Theorem 1 *If $(s, j_1, k_1, \ldots, j_{p-1}, k_{p-1}, j_p, t)$ is a shortest path from s to t in G_H then $H' = (H \cup \{j_1, j_2, \ldots, j_p\}) - \{k_1, k_2, \ldots, k_{p-1}\}$ is a valid UIO assignment for $dom(H')$ and $G'[H']$ is acyclic.*

Theorem 2 *If G_H has no path from s to t then H is a required solution for the BUAP.*

5 Algorithm for the Rural Postperson Problem

As stated earlier, given a strongly connected directed weighted graph $G = (V, E)$ and an edge subset FF of E, the RPP is to find a tour with minimum cost which traverses each edge in FF at least once. This problem is known to be NP-complete. In this section we present a heuristic algorithm, called *app_rpt* for the RPP.

Algorithm *app_rpt* repeatedly applies the rural symmetric augmentation algorithm of Aho *et al* [4]. We refer to this as *rural_symm_aug* (G, F, G_1, E_1). This algorithm accepts a weighted digraph $G = (V, E)$, and an edge set $F \subseteq E$ and computes a rural symmetric augmentation $G_1 = G[F \cup E_1]$ of G with respect to F by finding a minimum cost bag E_1 of edges from E such that G_1 is symmetric.

The algorithm *app_rpt* consists of three steps. The first step calls the algorithm *rural_symm_aug*(G, FF, G_0, E_0) to compute a rural symmetric augmentation G_0 of G with respect to the given set $FF \subseteq E$. If G_0 is weakly connected, then the algorithm outputs an euler tour of G_0 as the required tour and terminates. Otherwise it proceeds to the second step. This step joins the subtours in G_0 in an iterative fashion by computing rural symmetric augmentations of different auxiliary graphs with respect to certain subsets of FF. The idea of joining subtours is also applied by Frieze *et al* in their heuristic algorithm [16] for the asymmetric traveling salesperson problem [33, 26]. The third step further minimizes the cost of the tour obtained at the end of the second step. The formal description of the algorithm is given below.

```
Algorithm app_rpt(G, FF, Γ)
{ The algorithm finds an approximate RPT Γ of G with respect to FF, }
{ where G = (V, E) is a directed weighted graph }
{ each edge and FF ⊆ E. }

Step 1 {Initial rural symmetric augmentation }
  rural_symm_aug(G, FF, G0, E0);
  if (G0 is weakly connected) then begin
      Compute an euler tour Γ of G0 ;
      Stop
  end
  else begin
      Let C1, C2, ..., C|c| be the components of G0;
      T := FF ∪ E0; K := |c|;
      Compute all pair shortest paths in G
  end

Step 2 {Compute rural symmetric augmentations of auxiliary graphs }
  repeat
  { Construct an auxiliary weighted digraph G' = (V', E') }
      V' := ∅; E' := ∅;
      F' := ∅;
      Vf := ∅; Vt := ∅;
      for i := 1 to K do begin
          Choose an edge e = (vf, vt) ∈ FF ∩ Ci ;
          Add vf to Vf and to V';
          Add vt to Vt and to V';
          Add e to E' and to F';
          Associate the cost of e in E as the cost of e in E';
      end
      for each vt ∈ Vt do
          for each vf ∈ Vf such that (vf, vt) ∉ F' do begin
              Add an edge e' = (vt, vf) to E';
              Let the cost of e' be that of a shortest path from vt to vf in G;
          end
      Let G' = (V', E');
      rural_symm_aug(G', F', G*, E*) ;
      Let T' be the bag of all underlying edges in E for the edges in E* ∪ F';
      Add all the edges in T' to T;
      Let C1, C2, ..., Ch be the components of G*;
      K := h;
  until(K = 1)

Step 3 { Delete unwanted edges from T and compute the final tour }
  Construct an undirected graph G'' from G[T] by fusing the end vertices
  of each edge in FF and ignoring the orientation of the remaining edges;
  Compute an MST T'' of G'';
  Let F'' be the set of edges in E corresponding to the edges in T'';
```

rural_symm_aug$(G, FF \cup F'', \widehat{G}, \widehat{E})$;
Compute an euler tour Γ of $\widehat{G}$;
end *app_rpt*.

The time complexity of the algorithm is $O(m^2 \log n + |c|^4 \log |c|)$, where $|c|$ is the number of weakly connected components of the rural symmetric augmentation obtained in the first step of the algorithm [27]. Let $cost(X)$ denote the total cost of all the edges in bag X, considering each occurrence of an edge in X as being separate. We have proved in [27] that the cost of the tour Γ produced by our *app_rpt* algorithm is $(1+\lceil log|c|\rceil)$-*approximate*[2] to the cost of an RPT of G with respect to FF. We summarize this result in the following theorem.

Theorem 3 *Suppose that Γ_{opt} is an RPT of G with respect to FF and that Γ is the output given by algorithm* app_rpt. *Then Γ is a single tour containing each edge in FF at least once and $cost(\Gamma) \leq (1 + \lceil log|c|\rceil) cost(\Gamma_{opt})$, where $|c|$ is the number of weakly connected components of the rural symmetric augmentation obtained in the first step. That is, the cost of Γ is $(1 + \lceil log|c|\rceil)$-approximate to the cost of an RPT of G with respect to FF.*

6 Generalized UIO testing method

As pointed before, the existing UIO-based methods can be applied only to a subset of strongly connected protocols which are represented as FSMs having at least one UIO-sequence for each state. These methods, however, have the merit of selecting minimum length test sequences with high fault coverage [24]. In this section, we propose a generalized approach which can be applied to any protocol satisfying the above conditions. This method produces test sequences with varied level of optimality depending on the structure of the protocol and the UIO-sequences considered.

Our method is based on the BUAP and the RPP. The method is formally described in algorithm: *guio_test*. We assume that the protocol representation graph $G_s = (S, E)$ is strongly connected and each of its states has a nonempty set of UIO-sequences. As defined earlier, MU_i is a nonempty set of UIO-sequences for each state s_i. Let $MU = MU_1 \cup MU_2 \cup \ldots \cup MU_n$. Let $R \subseteq E \times MU$ such that $(e, u) \in R$ iff $end(e) = head(u)$. Let $G' = (S, E')$ where $E' = \{(start(e), tail(u); label(e)@u) | (e, u) \in R\}$. The generalized method starts by checking whether the MU-method can be applied to select a test sequence for the given protocol. If so, the method computes a minimum length test sequence in Step 1 using the MU-method and terminates. If the MU-method fails to find a solution, then the generalized method uses the UIO assignment B obtained in the MU-method to calculate an approximate RPT Γ_1 of $G_s + B$ with respect to B using the algorithm *app_rpt*. In Step 2, the method finds a minimum set of transitions BE and a valid UIO assignment H for BE such that the resulting test graph $G'[H]$ spans G' and $G'[H]$ has the minimum number of connected components. This is done by invoking the *basic_assignment* algorithm.

In order to minimize the length of the test sequence, a valid UIO assignment H' for the transitions in $E - BE$ and a minimal bag EP of transitions from E are computed in Step 3 such that the graph $G'' = G'[H \cup H'] + EP$ is symmetric. Note that each transition in EP is repeated in G'' as many times as they occur in EP. H' and EP are obtained by computing a minimum cost maximum flow f^* of a multi-stage flow graph $G_f = (V_f, E_f)$ whose

[2]The cost of a tour A is said to be *k-approximate* to that of a tour B if $\frac{cost(A)}{cost(B)} \leq k$, where $cost(A), cost(B) \geq 0$, $k \geq 1$ and $cost(B) \neq 0$.

Edge	lower bound	cost	capacity
(s, x_i)	0	0	# of out trans. from s_i in E
(x_i, y_j)	0	0	# of trans. from s_i to s_j in $E - BE$
(y_i, z_j)	0	$length(u)$: $head(u) = s_i \wedge tail(u) = s_j$	∞
(z_i, z_j)	0	1	∞
(z_i, t)	0	0	# of out trans. from s_i in E
(x_i, z_j)	1	0	1

Table 4: Parameters on the edges of the flow graph G_f

construction is described below. $V_f = \{s,t\} \cup V_x \cup V_y \cup V_z$ where $V_x = \{x_1, x_2, \ldots, x_n\}, V_y = \{y_1, y_2, \ldots, y_n\}$, and $V_z = \{z_1, z_2, \ldots, z_n\}$. $E_f = E_{sx} \cup E_{xy} \cup E_{yz} \cup E_{zz} \cup E_{zt} \cup E_{xz}$ where, $E_{sx} = \{(s, x_i) \mid 1 \leq i \leq n\}, E_{xy} = \{(x_i, y_j) \mid \exists e \in E - BE \wedge start(e) = s_i \wedge end(e) = s_j\}, E_{yz} = \{(y_i, z_j) \mid \exists u \in MU_i \wedge tail(u) = s_j\}, E_{zz} = \{(z_i, z_j) \mid \exists e \in E \wedge start(e) = s_i \wedge end(e) = s_j\}, E_{zt} = \{(z_i, t) \mid 1 \leq i \leq n\}$, and $E_{xz} = \{(x_i, z_j) \mid \exists (e, u) \in H \wedge start(e) = s_i \wedge tail(u) = s_j\}$. The lower bound, cost, and capacity assigned to the edges of G_f are shown in Table 4.

Assignment of UIO-sequences for the transitions in $E - BE$ is done using the optimum flow f^* in G_f. For instance, an unit flow from x_i to z_k through the vertex y_j indicates that the UIO-sequence $u \in MU_j$ with $tail(u) = s_k$ is to be assigned to a transition in $E - BE$ from s_i to s_j. This assignment H' is computed in Step 3. EP is obtained by adding a transition from s_i to s_j to EP as many times as the flow $f^*(z_i, z_j)$ along the edge $(z_i, z_j) \in E_{zz}$.

If G'' is connected, then G'' is eulerian and an euler tour Γ_2 of G'' is computed in Step 4. Otherwise, an approximate RPT Γ_2 of $G_s + F$ with respect to F is computed in this step using the heuristic algorithm *app_rpt*, where $F = H \cup H'$. The tour with minimum cost is chosen from Γ_1 and Γ_2 and a test sequence is obtained from this tour by concatenating the subsequences and/or labels of the transitions along this tour. The algorithm *guio_test* is described below.

Algorithm *guio_test*(G_s, MU, G', TS);
  Step 1
    Apply the MU-method;
    Let B be the UIO assignment computed in the MU-method;
    Let E_1 be a bag in E computed in the MU-method
    such that $G'[B] + E_1$ is symmetric;
    if $(G'[B] + E_1$ is connected$)$ **then**
    begin
      Obtain the test sequence TS by concatenating the labels
      of the edges along an euler tour Γ of $G'[B] + E_1$;
      stop
    end
    else *app_rpt*$(G_s + B, B, \Gamma_1)$
  Step 2
    basic_assignment(G_s, MU, G', H);
    $BE \leftarrow dom(H)$;
  Step 3
    Compute a minimum cost maximum flow f^* of $G_f(V_f, E_f)$;
    $H' \leftarrow \emptyset$;
    for each $e = (s_i, s_j; a/o) \in E - BE$ **do**
    begin
      $f^*(x_i, y_j) \leftarrow f^*(x_i, y_j) - 1$;
      Choose $(y_j, z_k) \in E_f$ such that $f^*(y_j, z_k) > 0$;
      $f^*(y_j, z_k) \leftarrow f^*(y_j, z_k) - 1$;
      Let $u \in MU_j$ such that $head(u) = s_j$ and $tail(u) = s_k$;
      Add (e, u) to H';

Transitions	UIO-sequence	Transitions	UIO-sequence
t1	t3	t2, t5	t1
t6, t7	t6		

Table 5: UIO-sequence assignment using min-cost flow

end
$F = \{(start(e), tail(u); label(e)@u) \mid (e,u) \in H \cup H'\}$;
$EP \leftarrow \emptyset$;
for each $(z_i, z_j) \in E_f$ such that $f^*(z_i, z_j) > 0$ **do**
Add a transition from s_i to s_j with minimum cost to EP $f^*(z_i, z_j)$ times ;
Step 4
if ($G'' = G'[H \cup H'] + EP$ is connected) **then**
Find the euler tour Γ_2 of G'';
else *app_rpt*$(G_s + F, F, \Gamma_2)$;
Obtain the test sequence TS by concatenating the labels of
the edges along the tour with minimum cost between Γ_1 and Γ_2;
end *guio_test*.

We would like to note that the multi-stage flow problem formulation used above is similar to the one given in [29] for assigning UIO-sequences to the transitions in E. While the assignment obtained in [29] may not result in a connected test graph, an optimal flow of our flow graph always yields a connected test graph whenever such a test graph exists. This is due to the fact that any optimal flow in our flow graph always subsumes the UIO assignment obtained as a solution to the BUAP in Step 2. It can be seen that the rural symmetric augmentations made in the first step of *app_rpt* is redundant, as far as the *guio_test* is concerned, since similar augmentation is already done in Step 1 or Step 3 of *guio_test*. We do not however modify algorithm *app_rpt* due to its generic application.

The algorithm takes at most $O(n^2m^2\nu^2 + c^4 \log c)$ time units. Here, $c = \max\{|c_1|, |c_2|\}$, where $|c_1|$ and $|c_1|$ are the number of weakly connected components of $G'[B] + E_1$ and G'', respectively. As before, ν denotes the maximum number of UIO-sequences in MU for any state. The level of optimality of the test sequence obtained by the generalized method is summarized in the following theorem. The Proof of the theorem directly follows from Theorem 3 and the algorithm *guio_test*.

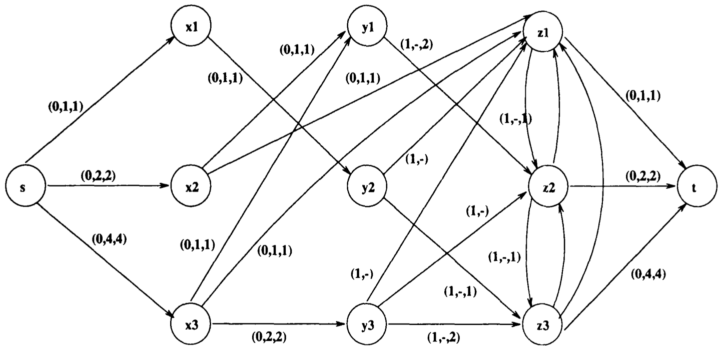
Theorem 4 *The length of the test sequence selected in the generalized method has the following levels of optimality.*

(i) if $G'[B] + E_1$ is connected then it is optimum

(ii) if G'' is connected then it is optimum subject to the condition that the edges in $dom(H)$ are preassigned using H, a solution to the BUAP.

(iii) In the worst case, it is always $(1 + \lceil \log(\min\{|c_1|, |c_2\}|)\rceil)$-approximate to the length of an optimal test sequence, where $|c_1|$ and $|c_2|$ are the number of connected components of $G'[B] + E_1$ and G'', respectively.

We now illustrate the proposed method on the FSM given in Figure 1. Let us consider the same sets of multiple UIO-sequences which are used in Section 3: $MU_1 = \{t1\}$, $MU_2 = \{t2, t3\}$, and $MU_3 = \{t4, t5, t6\}$. The generalized method finds a test sequence of minimum length (14) if $G'[B] + E_1$, computed in Step 1, is connected. If not, let us suppose that the UIO-assignment of the FSM obtained in Step 1 is A_1 as given in Table 2. Since $G'[A_1]$ itself is symmetric, $E_1 = \emptyset$. Since $G'[A_1] + E_1$ is not connected, *app_rpt* is



Note: (i) '-' indicates the infinite capacity (ii) Edge lables are the triplet (cost, capacity, flow), the last entry (flow) is omitted if it is zero (iii) (1, -) is assumed on all the unlabelled edges (iv) Assign a lower bound of one unit on edges (x2, z1) and (x3, z1) and zero for other edges

Figure 3: Flow graph for the FSM given in Figure 2

invoked in Step 1. Suppose that the set of edges $\{(s_1, s_1; t1\ t2), (s_2, s_3; t3\ t6)\} \subseteq E'$ is chosen as F' in the first iteration of the **repeat...until** loop of the algorithm *app_rpt* (Note that the label of the edges in this set is simply the sequence of underlying transitions in G_s). Then $T' = \{t1, t5\}$ and *app_rpt* moves to the third step. In Step 3 of *app_rpt*, the edge $t3$ is added so that $G'[A_1] + E_1$, $t1, t5$, and $t3$ together form the tour Γ_1: *t1 t2 t1 t2 t1 t3 t6 t5 t1 t3 t4 t3 t6 t6 t7 t6 t5* of length 17. The algorithm *basic_assignment* is invoked at Step 2 with the above set of multiple UIO-sequences. Suppose that the algorithm *basic_assignment* assigns the UIO-sequences $t5$ and $t2$ to transitions $t3$ and $t4$, respectively. Note that this assignment in fact yields a connected test graph. The multi-stage flow graph for computing the UIO assignment for the remaining transitions as well as a set of transitions to be added for obtaining G'' is shown in Figure 3. Labels in each edge is a triplet representing the cost, capacity, and the optimal flow, in that order. The last part of the triplet is omitted if the optimum solution has a zero flow along that edge. Edges (x_2, z_1) and (x_3, z_1) also have a unit lower bound. The resulting UIO assignment for the remaining transitions are shown in Table 5. The solution also indicates that $t1$ and $t3$ are the only additional transitions required for obtaining a rural symmetric augmentation of $G_s + (H \cup H')$ with respect to $H \cup H'$. *t1 t3 t6 t6 t7 t6 t5 t1 t2 t1 t3 t5 t1 t3 t4 t2* is the resulting tour Γ_2. Since the length of Γ_2 is less than the length of Γ_1, the test sequence is obtained by concatenating the input-output of the transitions along Γ_2. Observe that our generalized method produces a test sequence of length 16, two more than the optimum test sequence, whereas the MU-method by itself does not guarantee a test sequence.

Unlike the W-method [12] or the Wp-method [17], our method does not assume that the protocol has the reset capability. The fault coverage of this method with respect to output faults and transfer faults is same as that of the MU-method. Simulation studies show that the MU-method has very high fault coverage [24]. In [10], Chan *et al* have established that the UIO-based methods have complete fault coverage if the UIO-sequences are also UIO-sequences of the corresponding state in the IUT. The recent method by Yao *et al* [35] for protocols without reset capability selects test sequences which also include subsequences for verifying the UIO-sequences in the IUT. We would like to note that the inclusion of additional test subsequences for verification of the UIO-sequences increases the fault coverage at the price of considerable length increase in the test sequence.

7 Conclusions

The optimal UIO-based test selection methods (U-method and MU-method) [4, 30] do not cover certain protocols which are represented as strongly connected FSMs having at least one UIO-sequence for each state. In this paper we have generalized the MU-method so that it can be applied on any such protocol. Note that our approach does not require the reset capability. The method selects test sequences of different level of optimality depending on the structure of the protocol as well as the set of UIO-sequences used. The method uses solutions to the Basic UIO assignment Problem, and the Rural Postperson Problem. An efficient algorithm for the BUAP and a heuristic algorithm for the general RPP are also presented.

Since the length of the test sequence obtained in the last step of our method depends on the bound on the optimality of the solution obtained by the heuristic algorithm for the RPP, the length can be minimized further by designing better approximate algorithms for the RPP. To the best of our knowledge, our algorithm is the first heuristic for the asymmetric RPP with explicit bound on the optimality of the solution.

References

[1] CCITT/SGx/WP3-1, Specification and Description Language, SDL. CCITT Recommendations Z.100, 1988.

[2] ISO/IEC 9646: Information Processing Systems - Open Systems Interconnection - Conformance Testing Methodology and Framework, 1991.

[3] ISO SC21 WG1 P54: Information Processing Systems - Open Systems Interconnection - Formal Methods in Conformance Testing, Working Document, June 1993.

[4] A. V. Aho, A. T. Dahbura, D. Lee, and M. U. Uyar. An optimization technique for protocol conformance test generation based on UIO sequences and rural chinese postman tours. In S. Aggarwal and K. Sabnani, editors, *Protocol Specification, Testing and Verification, VIII*, pages 75–86. Elsevier Science Publishers B. V. (North-Holland), 1988.

[5] G. v. Bochmann, A. Das, R. Dssouli, M. Dubuc, A. Ghedamsi, and G. Luo. Fault models in testing. In *Proc. 4th International Workshop on Protocol Test Systems*, Leidschendam, The Netherlands, October 1991.

[6] G. v Bochmann and C. A. Sunshine. A survey of formal methods. In Green P. E., editor, *computer networks and protocols*, pages 561–578. Plenum Press, NewYork, USA, 1983.

[7] G.v. Bochmann, R. Dssouli, and J. R. Zhao. Trace analysis for conformance and arbitration testing. *IEEE Tr. Soft. Engg.*, SE-15:1347–1356, November 1989.

[8] T. Bolognesi and Ed. Brinksma. Introduction to the ISO specification language LOTOS. *Computer Networks and ISDN systems*, 14:25–59, 1987.

[9] S. Budkowski and P. Dembinski. An introduction to Estelle : A specification language for distributed systems. *Computer Networks and ISDN systems*, 14:3–23, 1987.

[10] W. Y. L. Chan, S. T. Vuong, and M. R. Ito. An improved protocol test generation procedure based on UIOs. In *ACM SIGCOMM*, pages 283–294, 1989.

[11] M. -S. Chen, Y. Choi, and A. Kershenbaum. Approaches utilizing segment overlap to minimize test sequences. In *Proc. 10th International Symposium on Protocol Specification, Testing and Verification*, pages 67–84, Ottawa, Canada, June 1990.

[12] T. S. Chow. Testing software design modeled by finite state machine. *IEEE Tr. Soft. Engg.*, SE-4(3):178–187, March 1978.
[13] A. Chung and D. Sidhu. Applications of sufficient conditions for efficient protocol test generation. In *Proc. 5th International Workshop on Protocol Test Systems*, pages 196–205, Montreal, Canada, September 1992.
[14] J. Edmonds. Matroid intersection. *Annals of discrete mathematics*, 4:39–49, 1979.
[15] J. Edmonds and E. L. Johnson. Matching, Euler Tours and the Chinese Postman. *Mathematical Programming*, 5:88–124, 1973.
[16] A. M. Frieze, G. Galbiati, and F. Maffioli. On the worst case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12:23–39, 1982.
[17] S. Fujiwara, G. v. Bochmann, F. Khendek, M. Amalou, and A. Ghedamsi. Test selection based on finite state models. *IEEE Tr. Soft. Engg.*, SE-17(6):591–603, June 1991.
[18] D. Hogrefe. OSI formal specification case study: the Inres protocol and service, Revised. Technical report, Institute for Informatics, University of Berne, May 1992.
[19] E. L. Lawler. Matroid intersection algorithms. *Mathematical programming*, 9:31–56, 1975.
[20] E. L. Lawler. *Combinatorial optimization : Networks and Matroids.* Holt, Reinhart and Winston, New York, USA, 1976.
[21] D. Y. Lee and J. Y. Lee. A well-defined Estelle specification for the automatic test generation. *IEEE Tr. Computers*, 40(4):526–542, April 1991.
[22] Z. Lidong, L. Jiren, and L. Huatian. A further optimization technique for conformance testing based on multiple UIO sequences. In *Proc. 5th International Workshop on Protocol Test Systems*, pages 206–211, Montreal, Canada, September 1992.
[23] R. E. Miller and S. Paul. Generating minimal length test sequences for conformance testing of communication protocols. In *Proc. IEEE INFOCOM*, April 1991.
[24] H. Motteler, A. Chung, and D. Sidhu. Fault coverage of UIO-based methods for protocol testing. In *Proc. 6th International workshop on protocol test systems*, pages 23–35, Pau, France, September 1993.
[25] C. H. Papadimitriou. On the complexity of edge traversing. *J. ACM.*, 23(3):544–554, July 1976.
[26] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Printice-Hall, New Jersey, USA, 1982.
[27] T. Ramalingam, K. Thulasiraman, and A. Das. A generalization of the multiple UIO method of test sequence selection for protocols represented in FSM. Technical report, Electrical Engineering, Concordia Univ, 1994.
[28] K. K. Sabnani and A. T. Dahbura. A new technique for generating protocol tests. In *Proc. 9th Data Communication Symposium*, pages 36–43. IEEE Computer Society press, September 1985.
[29] Y. -N. Shen and F. Lombardi. On two graph algorithms for the rural chinese postman tour problem in protocol verification and validation. Technical report, Department of Computer Science, Texas A & M University, College Station, U.S.A., 1992.
[30] Y. -N. Shen, F. Lombardi, and A. T. Dahbura. Protocol conformance testing using multiple UIO sequences. *IEEE Tr. Comm.*, 40(8):1282–1287, August 1992.
[31] D. P. Sidhu and T. -K. Leung. Formal methods for protocol testing: A detailed study. *IEEE Tr. Soft. Engg.*, SE-15(4):413–426, 1989.
[32] W. Stallings. *Networking Standards: a guide to OSI, ISDN, LAN, and MAN.* Addison-Wesley, New York, 1993.
[33] K. Thulasiraman and M.N.S. Swamy. *Graphs:Theory and algorithms.* John Wiley & sons, New York, USA, 1992.
[34] H. Ural. Formal methods for test sequence generation. *Computer Communications*, 15(5):311–325, June 1992.
[35] M. Yao, A. Petrenko, and G. v. Bochmann. Conformance testing of protocol machines without reset. In *Proc. 13th International Symposium on Protocol Specification, Testing and Verification*, 1993.