A Matroid-Theoretic Solution to an Assignment Problem in the Conformance Testing of Communication Protocols

T. Ramalingom, Krishnaiyan Thulasiraman, *Fellow*, *IEEE*, and Anindya Das

Abstract—The minimum length test sequence generation method proposed in [2] for conformance testing of a protocol uses Unique Input Sequences (UIS) for state identification. This method, called the U-method, requires that the test graph, a graph derived from the protocol, be connected. This requirement also needs to be satisfied in the case of the MU-method [31], [30], which assumes that the multiple UISs are available for each state. Thus, the U-method and the MU-method may not provide minimum length test sequences in cases where the test graph is not connected. Nevertheless, these methods generate minimum length test sequences with high fault coverage whenever the test graph is connected. This raises an important problem: Does there exist an assignment of UISs to the transitions such that the resulting test graph is connected? In this paper, we formulate this problem as a maximum cardinality two matroid intersection problem and discuss an efficient algorithmic solution. We also point out the role of the work in the minimum length test sequence generation problem.

Index Terms—Protocol, communication, communication protocol, protocol testing, graph theory, matroids, algorithms.

1 Introduction

Rapid and far-reaching advances in hardware and information technologies have resulted in complex distributed networks interconnecting heterogeneous computer systems to provide services such as data transfer and sharing of distributed resources. A protocol defines a set of rules for communication among these systems. Protocols are, in general, quite complex and, so, considerable efforts are required to implement them. The implementation of a protocol is based on a specification standard. A protocol standard, in general, can lead to several different implementations. Therefore, testing of a protocol for conformance to the specification of the protocol standard is called for. This testing is called conformance testing. As the services expected from a distributed system increase, the complexity of the communication protocols also increases, thus making conformance testing more challenging and most essential. Thus, protocol testing has become an integral part of the communication system design process.

Conformance testing involves selection of a test suite from the specification, execution of the test suite on the implementation under a specific test environment, and analysis of the test results. A current trend is to describe the specification of a protocol using one of the specification languages, such as Estelle, LOTOS, or SDL. Both control and data flow aspects of a protocol have to be tested to certify the implementation. The control flow part of a protocol can be represented as a deterministic Finite State Machine (FSM) and it can be derived from the protocol specification in Estelle, LOTOS, or SDL.

In the last decade or so, significant advances have been reported in the area of conformance testing, in particular, control flow testing using FSM methods. These include 1) techniques for minimizing the lengths of test sequences, 2) methods aimed at improving fault coverage, and 3) complexity results establishing intractability of certain problems. A detailed review of most of these methods may be found in [29], [6], [28], [25], [34], [1]. The recent work by Sun et al. [32] provides a comprehensive coverage of this area.

Different methods have been proposed in the literature for selecting test suites from an FSM specification. Test suite selected in these methods are simply a (test) sequence of input and expected output pairs. Ural [34] has reviewed various methods proposed in the past for selecting test sequences from the FSM specification of a protocol. As it is impractical to test the implementation exhaustively, these methods select only a test sequence of finite length using some selection criteria. Depending on the criteria, these methods select test sequences of varied lengths, fault detection capabilities (ability to detect the presence of a fault) and fault diagnosis capabilities (ability to localize the fault). An analysis of the fault detection and diagnosis capabilities of different test sequence selection methods may be found in [23].

Among the several test sequence selection methods, the U-method [26], [27], [2] has proven very popular and has been extensively studied. This method defines what is called a Unique Input-Output sequence (UIO sequence) of each state for state identification. A UIO sequence of a state is an input-output sequence of shortest length which lies along a walk starting from that state such that no walk

- *T. Ramalingom is with NORTEL Networks, PO Box 3511, Station C, Ottawa, Ontario K1Y 4H7, Canada.*
- *K. Thulasiraman and A. Das are with the School of Computer Science, University of Oklahoma, Norman, OK 73019. E-mail: thulasi@cs.ou.edu.*

Manuscript received 7 May 1997; revised 7 Dec. 1998; accepted 11 Feb. 2000. For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 105016.

starting from any other state has the same sequence. The problem of generating a minimum length test sequence using UIO sequences for state verification was formulated in [2] as an asymmetric Rural Postperson Problem (RPP) [9], [11], [33]. The efficient solution for RPP proposed in [2] requires a certain auxiliary graph, called the test graph, derived from the specification FSM for a given set of UIO sequences to be connected. In order to further minimize the length of the test sequence generated by the U-method, Shen et al. [31], [30] have proposed a method known as the MU-method. Instead of a single UIO sequence for each state, it uses multiple UIO sequences. The improvement is achieved by suitably assigning a UIO sequence for testing the tail state of a transition from the set of UIO sequences for that state such that the final test sequence is of minimum length. They have formulated the problem as an assignment problem which is solved using a minimum cost maximum flow algorithm. While attempting to generate a minimum length test sequence, the method in [31] also gives an assignment of UIO sequences for the transitions. The test graph resulting from this assignment of UIO sequences may not be connected. However, the method requires the test graph to be connected. But, there is no way to guide the minimum cost flow algorithm to produce an UIO sequence assignment leading to a connected test graph, even when one such assignment is available. Thus, the MU-method may not always lead to a minimum length test sequence.

The above discussion suggests an important problem. Does there exist an assignment of UIO sequences to the transitions such that the resulting test graph is connected? In this paper, we provide a matroid-theoretic formulation and solution of this problem. Our work is based on our earlier work presented in [24].

The rest of the paper is organized as follows: In Section 2, certain basic definitions relating to graphs, matroids, and finite state machines relevant to our work in this paper are presented. Section 3 discusses the U-method and the MU-method. Motivation for our work is presented in Section 4. In Section 5, a matroid theoretic formulation of the UIO assignment problem and an algorithmic solution are presented. In Section 6, we summarize our work, pointing out its role in solving the minimum length test sequence generation problem.

2 Basic Definitions

2.1 Graphs

The following definitions are taken from [33], [13], [20].

A **graph G** is a pair (V, E), where V and E are two finite disjoint sets. Elements of V and E are called vertices and edges, respectively. G is called an **undirected graph** if each edge is identified with an unordered pair of vertices referred to as the **end vertices** of the edge. An edge is called a **self-loop** if both its end vertices are identical. First, we shall consider an undirected graph G. An edge of G is said to be **incident** on its end vertices. A **walk** in a graph is a finite sequence of vertices $(v_0, v_1, \ldots, v_k)$, $k \geq 1$, such that (v_{i-1}, v_i) for $i = 1, 2, \ldots, k$ are edges in G. v_0 and v_k are called the **end vertices** of the walk and all other vertices are called its internal vertices. The walk $(v_0, v_1, \ldots, v_k)$ can also be denoted as a sequence of edges $(t_1, t_2, \ldots, t_k)$ if $t_i = (v_{i-1}, v_i)$, for $i = 1, 2, \ldots, k$. A walk is **open** if its end vertices are distinct; otherwise, it is **closed**. A closed walk is also referred to as a **tour**. A walk is called a **trail** if all its edges are distinct. An open trail is a **path** if all its vertices are distinct. A closed trail is called a cycle if all its vertices except the end vertices are distinct. A graph is said to be connected if there exists a path between every pair of vertices in the graph.

Weighted graphs are those graphs in which a real number, usually called the **cost**, is associated with each edge. The graph $G' = (V', E')$ is said to be a subgraph of the graph $G = (V, E)$ if $V' \subseteq V, E' \subseteq E$ and both end vertices of every edge in E' are in V'. A subgraph $G' = (V', E')$ of $G = (V, E)$ is called a **spanning subgraph** if $V' = V$. A subgraph $G' = (V', E')$ of $G = (V, E)$ is said to be **induced by** $F \subseteq E$ if $E' = F$ and V' is the set of end vertices of all the edges in F. This induced subgraph is denoted by $G[F]$. If F is a bag[1] of edges from E, then, in $G[F]$, each edge in F is repeated as many times as it occurs in F. If K is a set of edges having both their end vertices in V, then $G + K$ denotes the graph obtained from $G = (V, E)$ by adding all the edges in K to (the edge set of) G.

If each edge of a graph $G = (V, E)$ is identified with an ordered pair of vertices, then G is called a **directed graph** or, simply, a **digraph**. The edges in a digraph are also referred to as **arcs**. Let $a = (u, v) \in E$, where u and v are the vertices in V. a is called an **outgoing (incoming)** arc at $u(v)$. Also, u and v are called the **starting vertex** and the **tail vertex** of a, respectively. A walk in a digraph G is a finite sequence of vertices $(v_0, v_1, \ldots, v_k)$, $k \geq 1$, such that (v_{i-1}, v_i) for $i = 1, 2, \ldots, k$ are edges in G. v_0 and v_k are called the **starting vertex** and **tail vertex** of the walk, respectively, and all other vertices are called its internal vertices. A tour, a path, and a cycle in a digraph can be defined analogous to those in the undirected graph. A digraph is said to be **strongly connected** if there exists a path from each vertex to every other vertex. The digraph $G = (V, E)$ is said to be **symmetric** if the number of incoming arcs at every vertex in V is the same as the number of outgoing arcs at that vertex. Given a strongly connected digraph $G = (V, E)$ with weighted arcs, and a subset of arcs $F \subseteq E$, the **Asymmetric Rural Postperson Problem** (RPP) with respect to F is to find a tour with minimum cost such that it covers each arc in F at least once [9], [11], [33]. Such a minimum cost tour is referred to as an **Asymmetric Rural Postperson Tour** (RPT) with respect to F. The RPT is known to be an NP-complete problem [15], [19].

2.2 Matroids

Consider a finite set S of vectors over an arbitrary field. The collection of the independent sets of vectors possesses several interesting properties. For example,

1. Any subset of an independent set is independent.
2. If I_p and I_{p+1} are any two independent sets with $|I_{p+1}| = |I_p| + 1$, then I_p, together with some element of I_{p+1}, forms an independent set of $|I_{p+1}|$ elements.

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

It is interesting to note that there are several algebraic systems which possess the above properties. For instance, the collection of subsets of edges of a graph which do not contain any circuit possesses these properties. It was while studying the properties of such systems that Whitney [36] introduced the concept of a matroid. 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-1:** The empty set is a member of $\mathcal{I}$;
- **I-2:** $F_1 \subset F$ and $F \in \mathcal{I}$ imply $F_1 \in \mathcal{I}$;
- **I-3:** 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 $F_p \cup \{e\} \in \mathcal{I}$.

Elements of E are called the elements of the matroid M. Elements in $\mathcal{I}$ are called the **independent sets** in M. A maximal independent set of M is called a **base** of M. The **rank of the matroid** M is the cardinality of the largest independent set in M.

We now consider some examples.

Let S be a finite subset of a vector space. As noted above, the family of all the subsets of linearly independent vectors in S satisfy the independent axioms. Hence, these subsets of S form the collection of independent sets of a matroid on S. The rank of this matroid is equal to the dimension of the vector space S.

Another example of a matroid is the matching matroid defined on the vertex set of a graph. Consider an undirected graph G with vertex set V. A **matching** in G is a collection of edges of G such that no two edges have any common vertex. The vertices of these edges are said to be saturated in the matching [33]. Let $\mathcal{I}$ be the collection of all the subsets $I \subseteq V$ such that the elements of I are saturated in some matching of G. It can be shown that $\mathcal{I}$ is the collection of independent sets of a matroid on V. This matroid is called a **matching matroid**.

Let G be a connected undirected graph with edge set E. Consider the collection $\mathcal{I}$ of all the subsets of E which do not contain any cycle. Clearly $\mathcal{I}$ satisfies **I-1** and **I-2**. We can also show that $\mathcal{I}$ also satisfies **I-3** [33]. Thus, $\mathcal{I}$ is the collection of independent sets of a matroid M on E. Each base of M is a spanning tree of G. Therefore, in this matroid, the rank of E is equal to $n-1$, where n is the number of vertices of G. This matroid M is called a **graphic matroid**.

In this paper, two matroids are of special interest to us: the graphic matroid defined above and the partition matroid.

The partition matroid is defined as follows: 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 nonnegative 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 [13], [20]. It is called the **partition matroid** of G with respect to the partition P and the index Q.

Let $(E, \mathcal{I}_j)$, $j = 1, 2, \ldots, k$ for some $k \geq 2$, be a system of matroids over E. The **Maximum Cardinality Matroid Intersection Problem (MCMIP)** is to find a maximum subset H of E such that H is independent in $(E, \mathcal{I}_j)$ for $j = 1, 2, \ldots k$. If $k = 2$, then the above problem is referred to as the **Maximum Cardinality Two Matroid Intersection Problem (MC2MIP)**. Though the general MCMIP is NP-complete [21], [20], its special case MC2MIP is polynomially solvable [13].

Since its introduction in 1935 by Whitney, matroid has been extensively used to generalize seemingly different concepts and results in diverse disciplines such as electrical network theory, linear programming, graph theory, switching theory etc. It has helped to discover simpler and unifying proofs of several results in these disciplines by identifying the matroid structure underlying the problems under consideration. For example, Edmonds [7] showed that the greedy approach of Kruskal's minimum weight spanning tree problem applies to all optimization problems which seek to select a minimum weight element in a set, provided the set has the matroid structure. In the case of the minimum weight spanning tree problem, the set is the collection of all acyclic subgraphs of a graph. Several other matroid theoretic results, algorithms and applications may be found in [33], [13], [35], [18]. In this paper, we present yet another application of matroid in the area of communication protocol testing.

2.3 Finite State Machine Model of a Communication Protocol

As we mentioned earlier, the control flow part of a protocol can be modeled as an FSM. In the following, we shall highlight only those definitions relating to an FSM which are relevant to this paper. An FSM M can be formally defined as a 5-tuple $M = (S, s_1, I, O, T)$ where $S = \{s_1, s_2, \ldots, s_n\}$ is the nonempty finite set of states of M, s_1 is a designated state called the **initial state**, and I and O are nonempty finite sets of possible inputs and outputs of the protocol, respectively. The transition function T is a partial function defined as $T : S \times I \rightarrow S \times O$. $T(s_i, a) = (s_j, o)$ 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. If $t = ((s_i, a), (s_j, o)) \in T$, then t is called a transition in M. s_i and s_j are called the **starting state** and the **tail state** of t, respectively. We shall represent t as $(s_i, s_j; a/o)$. We often identify a state (s_i) in M by simply referring to its index (i).

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 $(s_i, s_j; a/o)$ corresponds to an arc in E directed from s_i to s_j with label $tid : a/o$, where tid is a unique transition identification for the transition/arc. The identification tid in the label of the transition is, in fact, optional. The graph G_s is called the specification graph of the corresponding protocol. Thus, we shall apply all graph theory definitions to the protocol as well. 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.

An input (output) sequence is a sequence of input (output) symbols. We denote an ordered pair (a, b) of input and output by a/b. An **input-output sequence** is a sequence of input and output pairs. "@" is an operator for concatenating two input (output) sequences, as well as input-output sequences. We shall denote the input sequence, the output sequence, and the input-output

sequence on a walk W by **Inseq(W)**, **Outseq(W)**, and **IOseq(W)**, respectively.

In FSM-based testing methods, the specifications and the implementations are assumed to be represented as FSMs. We shall refer to the FSM representations of a specification and an implementation of a protocol as SPEC and IUT, respectively. The methods also assume that the SPEC is strongly connected.

An input sequence U_i is called a **Unique Input Sequence (UIS)** of the state s_i of an FSM M if U_i is a sequence of shortest length such that 1) there exists a walk W from s_i such that $Inseq(i, W) = U_i$ and 2) for each state $s_j, j \neq i$, either there is no walk from s_j with input sequence U_i or $Outseq(j, U_i) \neq Outseq(i, U_i)$. Henceforth, $head(U_i)$ denotes the state s_i and $tail(U_i)$ denotes the tail state of W. Moreover, the input-output sequence IOseq(W) is called an **Unique Input Output (UIO) sequence** of s_i.

Let U_i be an UIS of s_i, $1 \leq i \leq n$. The set $\mathcal{U} = \{U_i \mid 1 \leq i \leq n\}$ is referred to as a **UIS set** of the FSM M.

In conformance testing, one is interested in minimization of test sequence lengths. So, we assign a unit cost to each edge in the specification graph $G_s = (S, E)$ of the protocol. We assume that the functions $start(e)$, $label(e)$, and $end(e)$ will return the starting state, label, and the tail state of any transition e, respectively. Let MU_i be a nonempty set of UISs for each state $s_i \in S$. Let $MU = MU_1 \cup MU_2 \cup \ldots \cup MU_n$. Define the relation[2] $R \subseteq E \times MU$ such that $(e, u) \in R$ iff $end(e) = head(u)$. Thus, for each transition e and each UIS u for the state $end(e)$, there is an element $(e, u) \in R$. Clearly, R denotes the set of all possible assignments of UISs from MU for all the transitions in E. We call any subset $B \subseteq R$ a **valid UIS assignment** or, simply, a **UIS assignment** for the set of transitions $D \subseteq E$ if $dom(B) = D$ and $|\{u \mid (e, u) \in B\}| = 1$, for each $e \in D$. That is, each element in D has exactly one UIS assigned in B. A valid UIS assignment for E is also referred to as a **(valid) UIS assignment of the protocol** G_s.

Consider the undirected graph $G' = (S, E')$, where $E' = \{(start(e), tail(u); label(e)@u) \mid (e, u) \in R\}$. An edge $e' \in E'$ which corresponds to $(e, u) \in R$ is often referred to as a **test edge** for the transition e since e can be tested by applying the sequence along e'. 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'. It is easy to see that there is a one-to-one correspondence between R and E'. An element of R is often treated as an edge in E' and vice versa. Let B be a valid UIS 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 UIS assignment of the protocol G_s is simply referred to as a **test graph for the protocol**. Observe that every test graph for a strongly connected protocol always spans all the states of the protocol. 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 since 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 since H is directed, the orientation of the edges in F' coinciding with that of the corresponding edges in G_s.

2. Given two sets A and B, the Cartesian product $A \times B$ is the set of all ordered pairs of elements $(x, y), x \in A, y \in B$. That is, $A \times B = \{(x, y) \mid x \in A, y \in B\}$. A subset $R \subseteq A \times B$ is called a relation from A to B. The domain of relation R, $dom(R) = \{x \mid (x, y) \in R\}$.

3 Test Sequence Selection Methods

Two important methods based on the UISs are described in this section. These methods have provided the motivation of the work presented in this paper. Recall that SPEC and IUT denote the FSM representation of a specification and an implementation of a protocol, respectively.

3.1 The U-Method

The U-method [2] [6] requires that the representation graph $G_s = (S, E)$ of SPEC be strongly connected. Each state of G_s is assumed to have an UIS. Let U_j be an UIS 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 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 UIS U_j is applied to the current state of the IUT and the output is examined against the expected output according to the SPEC.

Thus, the input sequence $a@U_j$ 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

$$E' = \{(s_i, tail(U_j); a@U_j) | (s_i, s_j; a/o) \in E\}.$$

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, an optimal test sequence for G_s lies along an RPT of G^* with respect to E'. In other words, the optimal test sequence generation 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 digraph $G = (V, E)$ with respect to $F \subseteq E$ is a digraph $G[F \cup E_1]$ such that 1) $G[F \cup E_1]$ is symmetric and 2) E_1 is a minimum cost bag in E satisfying 1). The polynomial algorithm given in [26] 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 containing elements in $E \cup E'$. A Euler tour of $G^*[E' \cup E_1]$ is a required RPT of G^* with respect to E'. The method generates a test sequence by concatenating the subsequences and/or inputs along a 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 [2]. 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.

3.2 MU-Method

In order to minimize the length of the test sequence, the multiple UIS-based methods [31], [30] choose one of many UISs available for each state. One such method proposed in [31] applies network flow techniques for assigning a suitable UIS for each transition from the set of multiple UISs of its tail state. However, this method has certain problems. The test graph resulting from the UIS assignment computed by the method may not be connected. In that

case, one needs heuristics to generate a test sequence to cover all the transitions. This also means that the method may not produce a minimum length test sequence if the resulting test graph is not connected. These problems are discussed in [30] and we shall refer to this modified method as the MU-method. This method generates a minimum length test sequence only when the underlying test graph for the UIS assignment resulting from the application of network flow techniques is connected.

Given a set MU_i of multiple UISs of minimum length for each state $s_i, i = 1, 2, \ldots n$ of the protocol $G_s = (S, E)$, the **UIS Assignment Problem (UAP)** is to find a valid UIS assignment B of the protocol such that the RPT of $G_s + B$ with respect to B is of minimum length among all valid UIS assignments of the protocol. The MU-method solves certain specific instances of this problem efficiently by transforming it into an equivalent multistage minimum cost maximum flow problem [30]. As in the U-method, a minimum length test sequence is obtained by concatenating the test subsequences and/or the inputs of the transitions along the minimum cost RPT. As we noted before, the MU-method guarantees an optimal test sequence for a protocol G_s if the test graph $G'[B]$ is connected. It has also been proven that protocols which have either the reset capability or a self-loop at each state always meet this requirement [2], [31]. As we shall see in the next section, this approach of obtaining minimum length test sequences may not work for all protocols.

An alternate approach for the test sequence length minimization problem in the presence of multiple UIO sequences is proposed in [34]. Techniques for minimizing the length of a test sequence by overlapping test subsequences of the transitions may be found in [4], [17], [14].

4 Motivation for the Present Work

It has been reported in [2] and [5] that the U-method can be applied to generate an optimum test sequence for a protocol G_s which satisfies one of the conditions 1 through 5 below. Note that conditions 1 through 4 are independent of UISs, whereas condition 5 is with respect to a particular UIS set.

1. G_s has the reset capability [2].

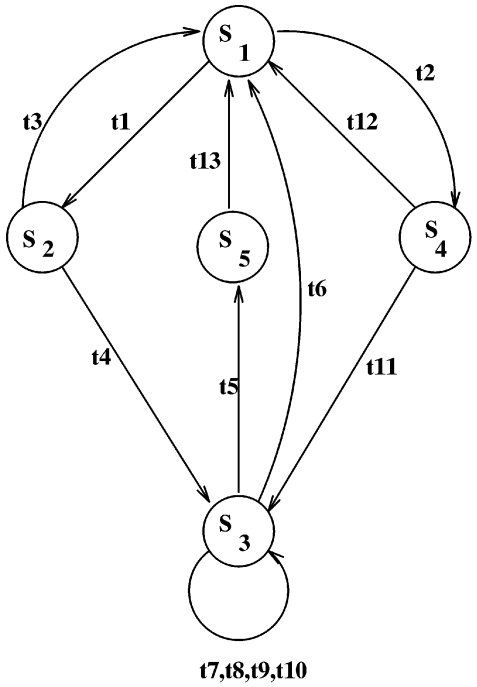


Fig. 1. Simplified transport protocol.

2. G_s has a self-loop at each state [2].
3. 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 [5].
4. 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 [5].
5. 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 UIS for the state s_j, $j = 1, 2, \ldots, n$, and it is used for testing every incoming transition at the state s_j [5].

We would like to note that the above conditions are only sufficient conditions. There are real-life protocols which do not satisfy any of these conditions, yet the U-method can

TABLE 1
Labels for Transitions in Fig. 1

Transition	Label	Transition	Label
t1	TCONreq/s-CR	t2	r-CR/TCONind
t3	r-DR/TDISind	t4	r-CC/TCONconf
t5	TDISreq/s-DR	t6	r-DR/s-DC&TDISind
t7	null/s-AK	t8	r-AK/null
t9	r-DT/TDATAind	t10	TDATAreq/s-DT
t11	TCONresp/s-CC	t12	TDISreq/s-DR
t13	r-DC/TDISconf		

TABLE 2
UISs for States of the FSM Shown in Fig. 1

State	UIS	State	UIS
S_1	t2 (r-CR)	S_2	t4 (r-CC)
S_3	t6 (r-DR)	S_4	t11 (TCONresp)
S_5	t13 (r-DC)		

successfully generate a minimum length test sequence for these protocols provided suitable UISs are chosen. For example, consider the FSM representation of a simplified transport protocol as given in [3] and shown in Fig. 1. The labels for the transitions are shown in Table 1. This protocol does not satisfy conditions 1-4. With the UISs generated as in Table 2, condition 5 is not met. However, the test graph of the protocol is connected if the UISs given in Table 2 are used. This example suggests that, even if a protocol does not have any of the structures stated in conditions 1-5, the U-method can still be used to obtain an optimum test sequence provided UISs resulting in a connected test graph are available.

Even when multiple UISs are available for all the states, careful assignment of UISs to transitions is necessary since an arbitrary assignment may not produce a connected test graph despite the existence of such assignments. For example, consider the abstract FSM protocol as given in Fig. 2, based on the *responder* module of the INRES protocol [10]. 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 UISs. The labels of the transitions are given in Table 3. Let $MU_1 = \{t1\}$, $MU_2 = \{t2, t3\}$, and $MU_3 = \{t4, t5, t6\}$ be the set of UISs for the states s_1, s_2, and s_3, respectively. Note that the UISs 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 4 and Table 5, respectively, are valid UIS assignments of the protocol. Also, both these assignments are obtainable in the MU-method while it attempts to solve the UAP. However, the test graph $G'[A_1]$ is not connected, whereas the other test graph $G'[A_2]$ is connected. If the UIS-based methods assign UISs to the transitions as per A_1, then they cannot generate a test sequence for this protocol. On the other hand, A_2 facilitates the UIS-based methods to generate an optimal test sequence for the protocol. The above discussion implies that the MU-method may not always produce a test sequence, even if the protocol has an UIS-set which yields a connected test graph. Unfortunately, there is no way to ensure that the min-cost flow approach used in these methods will lead to a test graph which is connected.

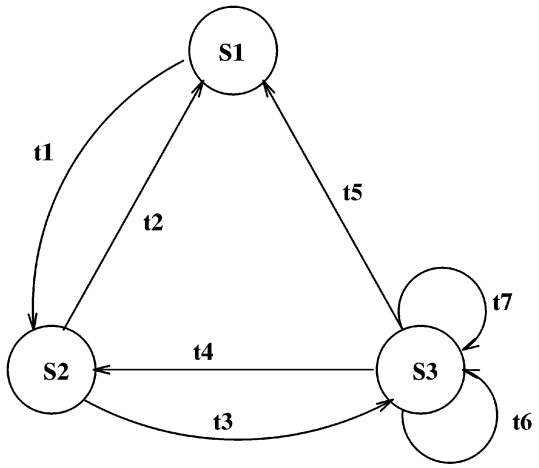


Fig. 2. An FSM based on the INRES protocol: *responder*.

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 Fig. 3. $m0$ and $m1$ are the only UISs for the states s_1 and s_2, respectively. The FSM neither satisfies the requirement stated in conditions 1 through 5 nor has a valid UIS assignment so that the resulting test graph is connected.

Thus, the following question arises: Given a set of multiple UISs for each state of a protocol, does the protocol have a set $BE \subseteq E$ of transitions and a valid UIS assignment for BE such that the resulting test graph for BE is connected and spans all the states of the protocol? If so, how to find a minimum set of transitions satisfying the above condition? In the following section, we formulate this problem as a Maximum Cardinality Two Matroid Intersection Problem (MC2MIP) and describe an algorithm to solve this problem.

TABLE 3
Labels of the Transitions in Fig. 2

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 4
Valid UIS Assignment without Connected Test Graph

Transition	UIS	Transition	UIS
t1	t2	t2, t5	t1
t3, t6, t7	t6	t4	t3

TABLE 5
Valid UIS Assignment with Connected Test Graph

Transition	UIS	Transition	UIS
t1	t3	t2, t5	t1
t3, t6, t7	t6	t4	t2

5 Basic UIS Assignment Problem

As defined earlier, let MU_i be a nonempty set of UISs for each state s_i of the strongly connected specification digraph $G_s = (S, E)$; $MU = MU_1 \cup MU_2 \cup \ldots \cup MU_n$. $R \subseteq E \times MU$ 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 UIS assignment B, the induced graph $G'[B]$ is a test graph for $dom(B)$.

The Basic UIS Assignment Problem (BUAP) is to find a maximum set $K \subseteq E$ and a valid UIS assignment B of K such that $G'[B]$ is acyclic and, hence, 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 MC2MIP which is solvable in polynomial steps. To start with, let us assume that G' is connected and that it has no self-loop. Let $M_1 = (E', \mathcal{I}_1)$ be the graphic matroid of G' defined in Section 2.2. Let Q_e be the set of all possible UIS assignments from MU for the transition e. Clearly, $Q_e \subseteq R$ and $dom(Q_e) = \{e\}$. Let $P = \{Q_e \mid e \in E\}$. Recall that an element $(e, u) \in R$ is also treated as an edge of 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 (see proof of Theorem 2). 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 for the MC2MIP by Lawler [12], [13] and Edmonds [8], as given in [18]. These algorithms are for computing a maximum cardinality intersection of any two matroids over the same set of elements. Algorithm *basic_assignment* is obtained from the above algorithms by adapting them for computing the maximum cardinality intersection of the graphic matroid M_1 and the partition matroid M_2 given above, thereby reducing the overall time complexity. The *basic_assignment* algorithm starts with an empty set of edges. That is, initially, $H = \emptyset$. At each iteration of the **repeat..until** loop of the algorithm, it computes a valid UIS 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 UIS assignment H output by the algorithm and $dom(H)$ form a solution to the BUAP. A formal description of the algorithm is given in Fig. 4. This is followed by an informal explanation. For the sake of simplicity in notation, we shall let an element $j = (e, u) \in E'$ also refer to the edge e.

An iteration of the **repeat...until** loop first (lines 5 - 18) 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 represents the edges in E' is partitioned into two sets: H and $E' - H$. Then, the graph G_H is constructed in such a way that the presence of a path from s to t in G_H guarantees that the cardinality of H in the current iteration can be increased by one. In order to construct the edge set E_H, the following is done with respect to 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 (lines 7 and 8). If $e \notin dom(H)$, then an edge from j to t is added to E_H (lines 9 and 10). 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 $e = e'$) (lines 13 and 14). 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 (lines 15 and 16).

If the digraph G_H thus constructed 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. Therefore, the algorithm proceeds to the next iteration of the **repeat...until** loop. 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 $dom(H)$ forms a solution to the BUAP (refer to Theorem 2).

Suppose that the cardinality of H computed in the current iteration is $n - 1$, where n is the number of states in G_s. Then, $G'[H \cup \{j\}]$ will have a cycle for each $j \in E' - H$. Therefore, G_H computed in the next iteration will not have any outgoing edge from s. In other words, G_H has no path from s to t. Also, the algorithm starts with H as an empty set and each iteration adds one edge to H. As a result, the algorithm terminates within n iterations.

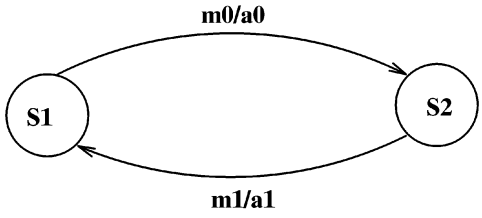


Fig. 3. Simplified alternating bit protocol.

```
Algorithm basic_assignment(G_s, MU, G', H);
{ Input: The digraph G_s = (S, E), graph G' = (S, E'), set of UISs MU }
{ Output: a set of edges H from E' }
1  H ← ∅ ;
2  V_H ← {s, t} ∪ E' ;
3  repeat
       {Construct the digraph G_H = (V_H, E_H)}
4      E_H ← ∅ ;
5      for each j = (e, u) ∈ E' − H do
6      begin
7          if (G'[H ∪ {j}] is acyclic) then
8              E_H ← E_H ∪ {(s, j)} ;
9          if (e ∉ dom(H)) then
10             E_H ← E_H ∪ {(j, t)} ;
11         for each k = (e', u') ∈ H do
12         begin
13             if (e = e') then
14                 E_H ← E_H ∪ {(j, k)} ;
15             if (G'[H ∪ {j}] has a unique cycle containing k) then
16                 E_H ← E_H ∪ {(k, j)} ;
17         end
18     end
19     if (the digraph G_H = (V_H, E_H) has a path from s to t) then
20     begin
21         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;
22         H ← (H ∪ {j_1, j_2, ..., j_p}) − {k_1, k_2, ..., k_{p−1}}
23     end
24     else
25     begin
26         output(H);
27         stop
28     end
29 for ever
30 end basic_assignment.
```

Fig. 4. The *basic_assignment* algorithm.

Let m and ν denote the number of transitions and the maximum number of UISs in MU for any state of the protocol, respectively. Since G_s is strongly connected, $m \geq n$, where n is the number of states in G_s. Suppose that the computation needed to check if a given set is independent in a given matroid is considered as one step. Then, our algorithm requires $O(n(m\nu)^2)$ steps. This complexity can easily be derived since the outer and the inner **for** loop of the **repeat..until** loop are executed at most $m \times \nu$ times and the **repeat..until** loop itself is executed at most n times. Note that this complexity is better than the complexity $(O(m\nu)^3))$ of the general maximum cardinality two-matroid intersection algorithms. When the time required to complete each step is

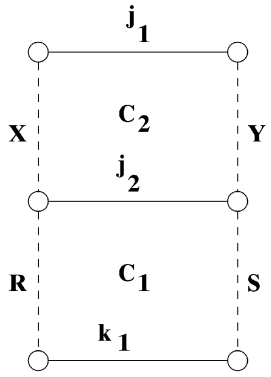


Fig. 5. Cycles in G'.

also taken into account, the complexity of *basic_assignment* becomes $O(n^2m^2\nu^2)$ time units. The following theorems establish the correctness of the algorithm.

Theorem 1. *Suppose that $H \subseteq E'$ is a valid UIS assignment and $G'[H]$ is acyclic at the beginning of a given iteration of the* **repeat..until** *loop of the basic_assignment algorithm. If $P = (s, j_1, k_1, \ldots, j_{p-1}, k_{p-1}, j_p, t)$ 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 UIS assignment for $dom(H')$ and $G'[H']$ is acyclic.*

Proof. We will first establish that $H' = (H \cup \{j_1, j_2, \ldots, j_p\}) - \{k_1, k_2, \ldots, k_{p-1}\}$ is a valid UIS assignment.

No pair of edges j_i and j_l, where $1 \leq i < l < p$, can be the UIS assignments for the same transition in E. Assume the contrary. Then, j_i and k_l will be UIS assignments for the same transition due to the fact that j_l and k_l assign UIS for the same transition. This means that $(j_i, k_l) \in E_H$ and that $(s, j_1, k_1, \ldots j_i, k_l, j_l + 1, k_l + 1, \ldots, j_{p-1}, k_{p-1}, j_p, t)$ is a path in G_H shorter than P; a contradiction.

Similarly, j_i and j_p, where $1 \leq i < p$, cannot be UIS assignments for the same transition. If this is not true, then let $e \in E$ be the transition for which j_i and j_p are UIS assignments. We know that $e \notin dom(H)$ because

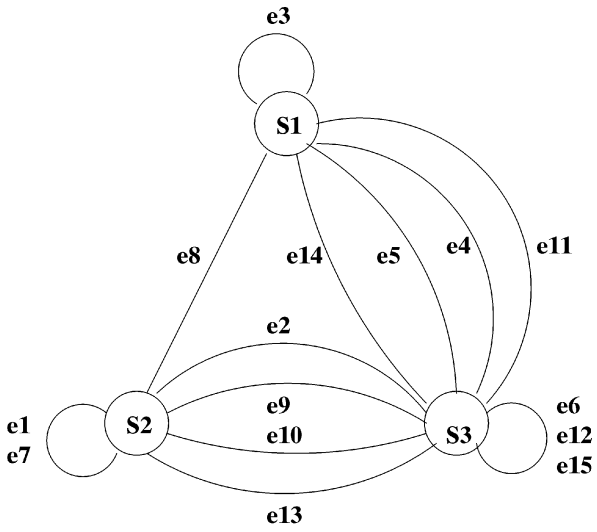


Fig. 6. G' for the FSM given in Fig. 2.

TABLE 6
Edge Description of G' for the FSM given in Fig. 2

Edge	Description	Edge	Description
$e1$	$(s_2, s_2; t2\ t1)$	$e2$	$(s_3, s_2; t5\ t1)$
$e3$	$(s_1, s_1; t1\ t2)$	$e4$	$(s_1, s_3; t1\ t3)$
$e5$	$(s_3, s_1; t4\ t2)$	$e6$	$(s_3, s_3; t4\ t3)$
$e7$	$(s_2, s_2; t3\ t4)$	$e8$	$(s_2, s_1; t3\ t5)$
$e9$	$(s_2, s_3; t3\ t6)$	$e10$	$(s_3, s_2; t6\ t4)$
$e11$	$(s_3, s_1; t6\ t5)$	$e12$	$(s_3, s_3; t6\ t6)$
$e13$	$(s_3, s_2; t7\ t4)$	$e14$	$(s_3, s_1; t7\ t5)$
$e15$	$(s_3, s_3; t7\ t6)$		

$(j_p, t) \in E_H$. So, $(j_i, t) \in E_H$. This means G_H has a path shorter than P; a contradiction.

Also, no $j_i, 1 \leq i < p$ and $h_m \in H$ ($h_m \neq k_l$ for any l) can both assign UIS for the same transition. If this is not so, then h_m and k_i will assign UIS for the same transition. It contradicts that H is a valid assignment.

From the above, it follows that $(H \cup \{j_1, j_2, \ldots, j_p\}) - \{k_1, k_2, \ldots, k_{p-1}\}$ is a valid UIS assignment.

Next, we show that $G'[H']$ is acyclic. The fact that $(s, j_1) \in E_H$ implies that $G[H_1]$ is acyclic, where $H_1 = H \cup \{j_1\}$. Let

$$H_i = H \cup \{j_1, j_2, \ldots, j_i\} - \{k_1, k_2, \ldots, k_{i-1}\},$$

for $2 \leq i \leq p$. We shall prove by induction that $G'[H_i]$ is acyclic, for $i = 2, 3, \ldots, p$. Note that $G'[H_p]$ is nothing but $G'[H']$. Since $(k_1, j_2) \in E_H$, k_1 is an edge in the unique cycle C_1 of $G'[H \cup \{j_2\}]$. As shown in Fig. 5, let us denote this cycle as $R\ j_2\ S\ k_1$, where R and S are some paths in G' whose edges are from H only.

We shall prove by contradiction that $G'[H_2]$ is acyclic. Suppose that $G'[H_2]$ has a cycle. Then, since $G'[H \cup \{j_2\} - \{k_1\}]$ and $G'[H \cup \{j_1\}]$ are both acyclic, j_1 and j_2 are in a cycle, say C_2, of $G'[H_2]$. Let $C_2 = X\ j_1\ Y\ j_2$, where X and Y are paths in G' whose edges are from $H - \{k_1\}$. This cycle is also shown in Fig. 5. Note that X, Y, R, and S may have common edges. Then, $C_1 \oplus C_2 \subseteq H_1$ has a cycle[3] [33], contradicting that $G'[H_1]$ is acyclic.

By assuming that $G'[H_i]$, where $2 \leq i < p$, is acyclic, we prove that $G'[H_{i+1}]$ is acyclic. Let

$$H_0 = H - \{k_1, k_2, \ldots, k_{i-1}\}.$$

Clearly,

$$H_i = H_0 \cup \{j_1, j_2, \ldots j_i\}$$

and

$$H_{i+1} = H_0 \cup \{j_1, j_2, \ldots j_{i+1}\} - \{k_i\}.$$

3. $C_1 \oplus C_2 = (C_1 \cup C_2) - (C_1 \cap C_2)$.

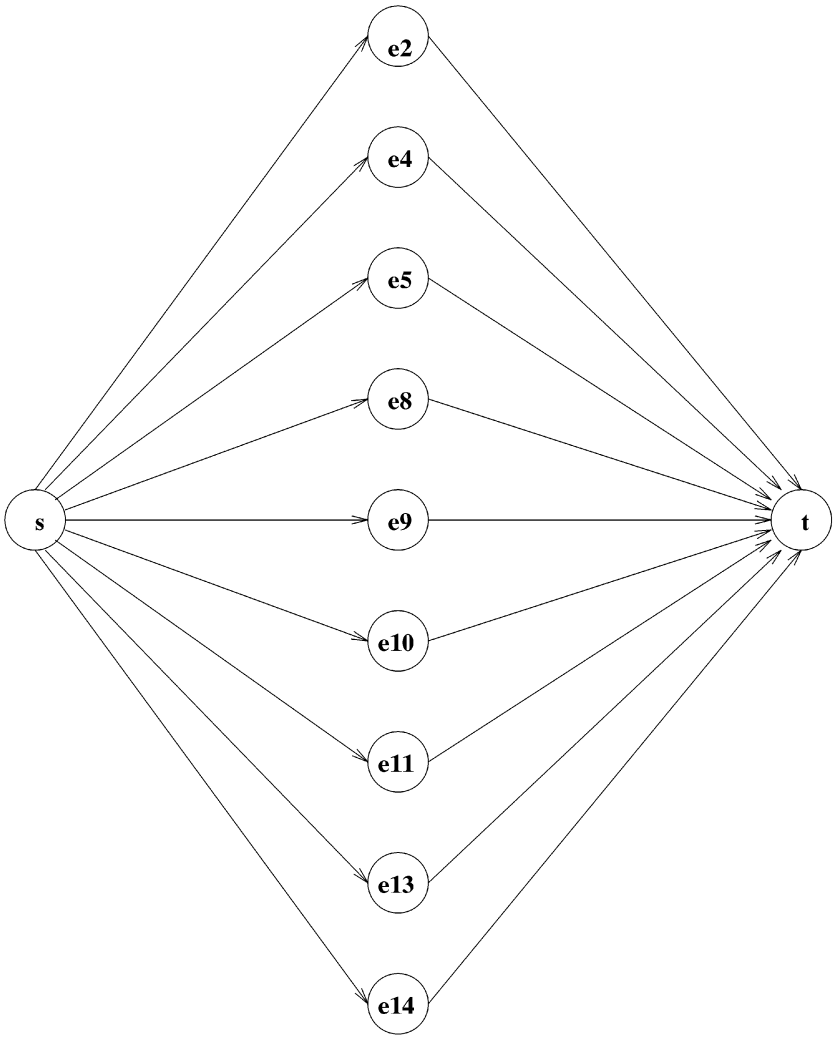


Fig. 7. G_H in the first iteration of *basic_assignment*.

Since $(k_i, j_{i+1}) \in E_H$, we know that k_i is contained in the unique cycle of $G'[H \cup \{j_{i+1}\}]$. Also, no edge from $\{k_1, k_2, \ldots, k_{i-1}\}$ is contained in this unique cycle of $G'[H \cup \{j_{i+1}\}]$. For, suppose on the contrary that k_r, for some r, $1 \leq r \leq i-1$, is an edge in the cycle, then $(k_r, j_{i+1}) \in E_H$ and $(s, j_1, k_1, \ldots, k_r, j_{i+1}, k_{i+1}, \ldots, j_p, t)$ is a path in G_H shorter than P. This is a contradiction since P is a shortest path in G_H.

Let C be the unique cycle in $G'[H \cup \{j_{i+1}\}]$. As we have just shown, $C \subseteq H_0 \cup \{j_{i+1}\}$. Also, by the induction hypothesis, we know that $G'[H_0 \cup \{j_1, j_2, \ldots j_i\}]$ is acyclic. Therefore, $G'[H_0 \cup \{j_1, j_2, \ldots j_i, j_{i+1}\} - \{k_i\}]$ is acyclic. For, otherwise, there is a cycle $C' \subseteq H_0 \cup \{j_1, j_2, \ldots j_{i+1}\}$ and $C \oplus C' \subseteq H_0 \cup \{j_1, j_2, \ldots j_i\}$ contains a cycle; a contradiction. This completes the induction and the proof of the theorem. □

At the beginning of the first iteration of the **repeat..until** loop of the algorithm *basic_assignment*, H is a valid assignment and $G'[H]$ is acyclic since $H = \emptyset$. At the start of a subsequent iteration, H corresponds to H' of the previous iteration. Therefore, as per Theorem 1, we know that, at the end of the **repeat..until** loop, H is a valid UIS assignment and $G'[H]$ is acyclic.

Theorem 2. *At the end of a given iteration of the* **repeat..until** *loop of the algorithm basic_assignment, if G_H has no path from s to t, then H and $dom(H)$ form a solution to the BUAP.*

Proof. In order to prove the theorem, we define the functions t and a from the power set of E' to the set of natural numbers including zero and the functions T and A from the powerset of E' to itself. Let X be a subset of E'.

$$t(X) = \text{number of edges in the largest acyclic subgraph of } G'[X],$$

$$a(X) = \text{number of elements in a maximum valid UIS assignment in } X,$$

$$T(X) = F, \text{ if } F \text{ is a largest superset of } X \text{ in } E' \text{ such that } t(F) = t(X),$$

$$A(X) = F, \text{ if } F \text{ is a largest superset of } X \text{ in } E' \text{ such that } a(F) = a(X).$$

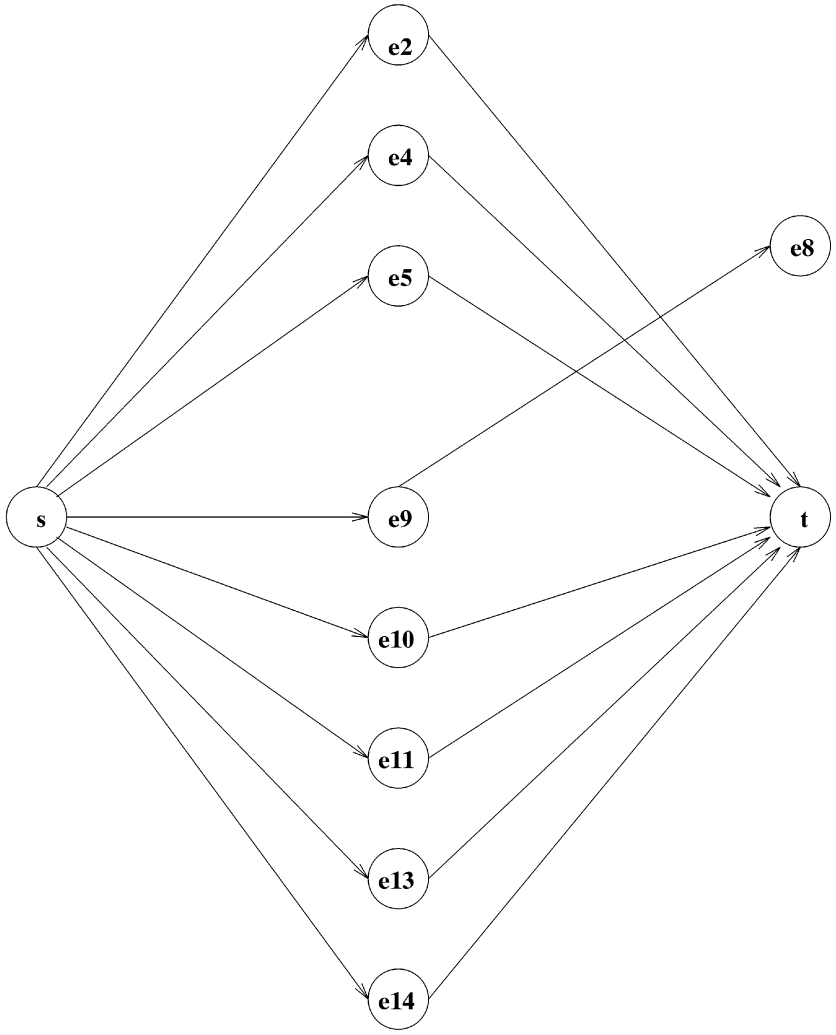


Fig. 8. G_H in the second iteration of *basic_assignment*.

We will first establish that H is a maximum valid UIS assignment in E' such that $G'[H]$ is acyclic. Let R denote the set of all vertices in G_H which are reachable from s. Formally, $R = \{v \in E' \mid \exists \text{ a path from } s \text{ to } v \text{ in } G_H\}$. Let $NR = E' - R$, $RH = R \cap H$, and $NRH = NR \cap H$.

We claim that $R \subseteq A(RH)$. RH is a valid UIS assignment since its superset H itself is a valid UIS assignment. So, $RH \subseteq A(RH)$. Let $j \in R - RH$. In order to prove our claim, it is enough if we prove that RH is a maximum valid UIS assignment in $RH \cup \{j\}$. In other words, we have to prove that $RH \cup \{j\}$ is not a valid UIS assignment. Let j be a UIS assignment for the transition e. Suppose that $RH \cup \{j\}$ is a valid UIS assignment, then $e \notin dom(RH)$. Also, $e \notin dom(H - R)$ for, otherwise, an element in $H - R$ will be reachable from s. Therefore, $e \notin dom(H)$ and $(j, t) \in E_H$. But, then, there exists a path from s to t in G_H because $j \in R$. This is a contradiction.

Our next claim is that $NR \subseteq T(NRH)$. Since $NRH \subseteq T(NRH)$, it is enough if we prove that $NR - NRH \subseteq T(NRH)$. Let $j \in NR - NRH$. Note that $G'[NRH]$ is acyclic. We have to prove that the largest acyclic subgraph of $G'[NRH \cup \{j\}]$ is $G'[NRH]$. If not, then $G'[NRH \cup \{j\}]$ is acyclic. But, $G'[H \cup \{j\}]$ has a cycle because $j \notin R$ and, so, $(s, j) \notin E_H$. Therefore, there exists a $k \in RH$ such that it is contained in the unique cycle in $G'[H \cup \{j\}]$. Then, $(k, j) \in E_H$. $k \in RH$ and $(k, j) \in E_H$ together imply $j \in R$. This is a contradiction since $j \in NR - NRH$.

Let H_{max} be a maximum valid UIS assignment in E' such that $G'[H_{max}]$ is acyclic. Clearly,

$$|H| \leq |H_{max}|. \tag{1}$$

The following derivation is obtained using the above claims.

$$\begin{aligned} |H_{max}| &= |H_{max} - R| + |H_{max} \cap R| \\ &= t(H_{max} - R) + a(H_{max} \cap R) \\ &\leq t(E' - R) + a(R) \\ &= t(NR) + a(R) \\ &\leq t(T(NRH)) + a(A(RH)) \\ &= |NRH| + |RH| \\ &= |H|. \end{aligned} \tag{2}$$

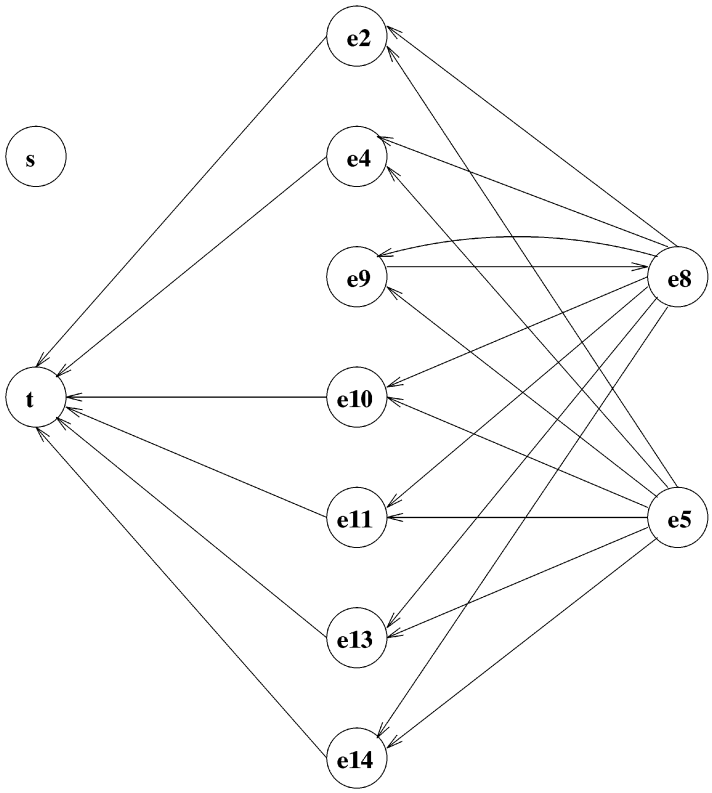


Fig. 9. G_H in the third iteration of *basic_assignment*.

Combining (1) and (2), we conclude that H is a maximum valid UIS assignment in E' such that $G'[H]$ is acyclic.

Suppose H does not span all the vertices in G'. Let v be a vertex in G' which is not spanned by H. Let $e' \in G'$ be an edge incident at v such that $e' = (e, u) \in R$ and $start(e) = v$. Clearly, $H \cup \{e'\}$ is a valid UIS assignment and $G'[H \cup \{e'\}]$ is acyclic. This contradicts that H is a maximum valid UIS assignment. Therefore, H spans all the vertices is G'.

We shall prove by contradiction that $G'[H]$ has the minimum number of connected components. Suppose it is not true. Then, let P be an UIS assignment such that $G'[P]$ is acyclic, spans every vertex in G', and has the minimum number of connected components. Let p and q be the number of connected components in $G'[H]$ and $G'[P]$, respectively. Clearly, $p > q$. Since $G'[H]$ and $G'[P]$ are both acyclic, it follows that the number of edges in H and P are $n - p$ and $n - q$, respectively. Since $p > q$, we have $n - p < n - q$. That is, H has fewer number of edges than P. This is a contradiction since H is a maximum valid UIS assignment with $G'[H]$ acyclic. In other words, $G'[H]$ has the minimum number of connected components. Thus, we have proven that H is a maximum valid UIS assignment such that $G'[H]$ is acyclic. In other words, H and $dom(H)$ provide a solution to the BUAP.□

In the presentation of the solution to the BUAP, we have assumed that G' is connected and it has no self-loop. Our approach works for the general case as well. Suppose that the graph obtained from G' by removing all the self-loops is not connected. Let $C_1, C_2, \ldots, C_p$ be the connected components of the resulting graph, where $p \geq 2$. Let K_i and B_i be the solutions for the BUAP for C_i, where K_i and B_i denote the edge set and the UIS assignment for K_i, respectively. If C_i is just a single vertex, say s_k, then $B_i = \{e'\}$ and $K_i = \{e\}$, where $e' = (e, u) \in R$ is some self-loop at s_k in G'. If C_i is not a single vertex, then its solution is obtained using the algorithm $basic\_assignment$. Then, it is easy to see that $K = K_1 \cup K_2 \cup \ldots \cup K_p$ and $B = B_1 \cup B_2 \cup \ldots \cup B_p$ form a solution to the original BUAP.

As an illustration for the *basic_assignment* algorithm, we consider the FSM as given in Fig. 2 with $MU_1 = \{t1\}$, $MU_2 = \{t2, t3\}$, and $MU_3 = \{t4, t5, t6\}$ as the sets of multiple UISs for the states s_1, s_2, and s_3, respectively. Note that we have referred to a UIS of a state by the corresponding transition along which the sequence lies. The graph $G' = (S, E')$ for the FSM is given in Fig. 6. Here, $E' = \{e1, e2, \ldots e15\}$. The edges are described in Table 6. For convenience, we represent the label of an edge in E' as the sequence of transitions along which the label of the edge lies. As discussed before, the *basic_assignment* algorithm considers only the non-self-loop edges in G'. Thus, the set $\{e1, e3, e6, e7, e12, e15\}$ of edges are removed from E'.

To start with, $H = \emptyset$. The first part of the **repeat...until** loop constructs a digraph $G_H = (V_H, E_H)$, where $V_H = \{s, t\} \cup E'$. E_H is the empty set at the beginning of the first iteration. The edge set E_H after completely executing the first **for** loop is shown in Fig. 7 as part of the digraph G_H. Observe that, for instance, an edge from s to $e8$ was added to E_H in this loop since $G'[H \cup \{e8\}]$ is acyclic (see lines 7 and 8 in the algorithm). Also, an edge

from $e8$ to t was added to E_H since the transition $t3$, for which $e8$ is a test edge, does not obviously belong to $dom(H) = \emptyset$ (lines 9 and 10). Since $(s, e8, t)$ is a shortest path from s to t in G_H, $e8$ is added to H (lines 21 and 22). Therefore, $H = \{e8\}$ and $dom(H) = \{t3\}$. The algorithm enters into the second iteration of the **repeat...until** loop.

The digraph G_H constructed in the second iteration of the **repeat...until** loop is shown in Fig. 8. Note that an edge from $e9$ to $e8$ is present in G_H, since $e9$ and $e8$ are test edges for the same transition $t3$ (lines 13 and 14). Since $(s, e5, t)$ is a shortest path from s to t in the current G_H, $e5$ is added to H. Thus, $H = \{e8, e5\}$ and $dom(H) = \{t3, t4\}$ at the end of the second iteration of the **repeat...until** loop. Fig. 9 shows the graph G_H constructed in the third iteration of the loop. Observe that the edge from $e5$ to $e2$, for instance, is present in G_H since $e5$ and $e2$ are contained in the unique cycle in $G'[H \cup \{e2\}]$ (lines 15 and 16). The algorithm terminates in this iteration since there is no path from s to t in G_H. Thus, the solution to the basic assignment problem at hand is $H = \{e8, e5\}$ with $dom(H) = \{t3, t4\}$. That is, assign the UIS along $t5$ to the transition $t3$ and the UIS along $t2$ to the transition $t4$. Observe that $G'[H]$ is a spanning tree of G'.

6 Summary

The minimum length test sequence generation method proposed in [2] for conformance testing of a protocol uses unique input sequences for state identification. This method, called the U-method, requires that the test graph be connected. This requirement also needs to be satisfied in the case of the MU-method [31], [30], which assumes that the multiple UISs are available for each state. In other words, the U-method and the MU-method do not cover certain protocols. Nevertheless, these methods generate minimum length test sequences with high fault coverage [16] whenever the test graph is connected. This raises an important problem: Does there exist an assignment of UIS to the transitions such that the resulting test graph is connected? In this paper, we have formulated this problem as a maximum cardinality two matroid intersection problem and discussed an efficient algorithmic solution.

Our work suggests the following approach for the application of UIS-based methods for minimizing the length of the test sequence: 1) Apply the MU-method and if it results in a connected test graph, then a minimum length test sequence can be generated efficiently. 2) If the application of the MU-method does not result in a connected test graph, then apply our BUAP algorithm (Section 5) to obtain a UIS assignment that results in a connected test graph (if such an assignment exists). A minimum length test sequence for this UIS assignment can be generated efficiently. 3) If the application of the BUAP algorithm does not lead to a connected test graph, then apply the heuristics presented in [24], [22] to find test sequences of different levels of optimality. These heuristics involve a minimum cost spanning tree algorithm, an approximate algorithm for the asymmetric rural postperson problem, and network flow techniques. These results will be discussed in a subsequent paper.

Acknowledgments

This work was done at Concordia University, Montreal, prior to T. Ramalingom joining NORTEL Networks and represents the views of the authors and not necessarily those of NORTEL Networks.

References

[1] R. Anido and A. Cavalli, "Guaranteeing Full Fault Coverage for UIO Based Methods," *Proc. Eighth Int'l Workshop Protocol Test Systems,* Sept. 1995.

[2] 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," *Protocol Specification, Testing and Verification, VIII,* S. Aggarwal and K. Sabnani, eds., pp. 75-86, North-Holland: Elsevier Science Publishers B.V., 1988.

[3] G. v. Bochmann, R. Dssouli, and J.R. Zhao, "Trace Analysis for Conformance and Arbitration Testing," *IEEE Trans. Software Eng.,* vol. 15, no. 11, pp. 1,347-1,356, Nov. 1989.

[4] M.-S. Chen, Y. Choi, and A. Kershenbaum, "Approaches Utilizing Segment Overlap to Minimize Test Sequences," *Proc. 10th Int'l Symp. Protocol Specification, Testing, and Verification,* pp. 67-84, June 1990.

[5] A. Chung and D. Sidhu, "Applications of Sufficient Conditions for Efficient Protocol Test Generation," *Proc. Fifth Int'l Workshop Protocol Test Systems,* pp. 196-205, Sept. 1992.

[6] A.T. Dahbura, K.K. Sabnani, and M.U. Uyar, "Formal Methods for Generating Protocol Conformance Test Sequences," *Proc. IEEE,* vol. 78, no. 8, pp. 1,317-1,326, 1990.

[7] J. Edmonds, "Matroids and Greedy Algorithm," *Math. Programming,* vol. 1, pp. 127-136, 1971.

[8] J. Edmonds, "Matroid Intersection," *Annals of Discrete Math.,* vol. 4, pp. 39-49, 1979.

[9] J. Edmonds and E.L. Johnson, "Matching, Euler Tours and the Chinese Postman," *Math. Programming,* vol. 5, pp. 88-124, 1973.

[10] D. Hogrefe, "OSI Formal Specification Case Study: The Inres Protocol and Service, Revised," technical report, Inst. for Informatics, Univ. of Berne, May 1992.

[11] M.-K. Kuan, "Graphic Programming Using Odd or Even Points," *Chinese Math.,* vol. 1, pp. 273-277, 1962.

[12] E.L. Lawler, "Matroid Intersection Algorithms," *Math. Programming,* vol. 9, pp. 31-56, 1975.

[13] E.L. Lawler, *Combinatorial Optimization: Networks and Matroids.* New York: Holt, Reinhart and Winston, 1976.

[14] Z. Lidong, L. Jiren, and L. Huatian, "A Further Optimization Technique for Conformance Testing Based on Multiple UIO Sequences," *Proc. Fifth Int'l Workshop Protocol Test Systems,* pp. 206-211, Sept. 1992.

[15] J.K. Lenstra and A.H.G. Rinnooy Kan, "On General Routing Problems," *Networks,* vol. 6, no. 3, pp. 273-280, July 1976.

[16] H. Motteler, A. Chung, and D. Sidhu, "Fault Coverage of UIO-Based Methods for Protocol Testing," *Proc. Sixth Int'l Workshop Protocol Test Systems,* pp. 23-35, Sept. 1993.

[17] R.E. Miller and S. Paul, "Generating Minimal Length Test Sequences for Conformance Testing of Communication Protocols," *Proc. IEEE INFOCOM,* Apr. 1991.

[18] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization.* New York: John Wiley & Sons, 1988.

[19] C.H. Papadimitriou, "On the Complexity of Edge Traversing," *J. ACM,* vol. 23, no. 3, pp. 544-554, July 1976.

[20] R.G. Parker and R.L. Rardin, *Discrete Optimization.* San Diego, Calif.: Academic Press, 1988.

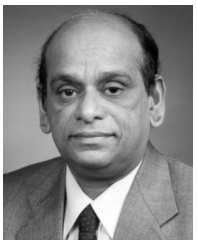
[21] C.H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity.* Englewood Cliffs, N.J.: Prentice Hall, 1982.

[22] T. Ramalingam, "Test Case Generation and Fault Diagnosis Methods for Communication Protocols Based on FSM and EFSM Models," PhD thesis, Concordia Univ., Montreal, Canada, 1994.

[23] T. Ramalingam, A. Das, and K. Thulasiraman, "Fault Detection and Diagnosis Capabilities of Test Sequence Selection Methods Based on the FSM Model," *Computer Comm.,* vol. 18, no. 2, pp. 113-122, Feb. 1995.

[24] T. Ramalingam, K. Thulasiraman, and A. Das, "A Generalization of the Multiple UIO Method of Test Sequence Selection for Protocols Represented in FSM," *Proc. Seventh Int'l Workshop Protocol Test Systems,* Nov. 1994.
[25] M. Rodrigues and H. Ural, "Optimal Length Test Sequences: Lower Bounds on Their Length and Exact Solutions for Their Construction," technical report, computer science, Univ. of Ottawa, Ottawa, Canada, 1992.
[26] K.K. Sabnani and A.T. Dahbura, "A New Technique for Generating Protocol Tests," *Proc. Ninth Data Comm. Symp.,* pp. 36-43, Sept. 1985.
[27] K. Sabnani and A. Dahbura, "A Protocol Test Generation Procedure," *Computer Networks and ISDN Systems,* vol. 15, pp. 285-297, 1988.
[28] D. Sidhu, "Protocol Testing: The First Ten Years, the Next Ten Years," *Proc. 10th Int'l IFIP Symp. Protocol Specifications, Testing, and Verification,* June 1990.
[29] D.P. Sidhu and T.-K. Leung, "Formal Methods for Protocol Testing: A Detailed Study," *IEEE Trans. Software Eng.,* vol. 15, no. 4, pp. 413-426, Apr. 1989.
[30] Y.-N. Shen and F. Lombardi, "On Two Graph Algorithms for the Rural Chinese Postman Tour Problem in Protocol Verification and Validation," technical report, Dept. of Computer Science, Texas A&M Univ., College Station, Tex., 1992.
[31] Y.-N. Shen, F. Lombardi, and A.T. Dahbura, "Protocol Conformance Testing Using Multiple UIO Sequences," *IEEE Trans. Comm.,* vol. 40, no. 8, pp. 1,282-1,287, Aug. 1992.
[32] X. Sun, Y. Shen, C. Feng, and F. Lombardi, *Protocol Conformance Testing Using Unique Input/Output Sequences.* World Scientific, 1998.
[33] K. Thulasiraman and M.N.S. Swamy, *Graphs: Theory and Algorithms.* New York: John Wiley & Sons, 1992.
[34] H. Ural, "Formal Methods for Test Sequence Generation," *Computer Comm.,* vol. 15, no. 5, pp. 311-325, June 1992.
[35] D.B. West, *Introduction to Graph Theory.* Prentice Hall, 1996.
[36] H. Whitney, "On the Abstract Properties of Linear Dependence," *Am. J. Math.,* vol. 57, pp. 509-533, 1935.

T. Ramalingam received the MSc degree in mathematics from Aditanar College of the Madurai Kamaraj University, India, the MTech degree in computer science and data processing from the Indian Institute of Technology, Kharagpur, India, and the PhD degree in electrical and computer engineering from Concordia University, Montreal, Canada, in 1983, 1988, and 1994, respectively. He is currently a senior software designer at the Access Network Division of Nortel Networks, Ottawa, Canada. Prior to joining the PhD program at Concordia, he had worked as a senior scientific officer in the Department of Computer Science and Engineering of the Indian Institute of Technology, Madras. His research interests include various aspects of communication protocol engineering and data base systems.



Krishnaiyan Thulasiraman holds the Hitachi Chair and is a professor in the School of Computer Science at the University of Oklahoma, Norman, where he has been since 1994. He received his Bachelor's degree (1963) and his Master's degree (1965), both in electrical engineering, from the University of Madras, India, and his PhD degree (1968) in electrical engineering from the Indian Institute of Technology, Madras. Prior to joining the University of Oklahoma, Dr. Thulasiraman was a professor (1981-1994) and chair (1993-1994) of the Electrical and Computer Engineering Department at Concordia University, Montreal, Canada. He was on the faculty in the Electrical Engineering and Computer Science Departments of the Indian Institute of Technology, Madras, during 1965-1981 and served as a professor since 1977. His research interests have been in graph theory, combinatorial optimization, and algorithms with applications in a variety of areas in computer science and electrical engineering: electrical networks, VLSI physical design systems level testing, communication protocol testing, parallel/distributed computing telecommunications network planning, interconnection networks, etc. Dr. Thulasiraman has published extensively in these areas, and has coauthored two text books *Graphs, Networks and Algorithms* (1981) and *Graphs: Theory and Algorithms* (1992), both published by Wiley Interscience, and two chapters in the *Handbook of Circuits and Filters* (CRC and IEEE Press, 1995).

Dr. Thulasiraman has received several awards and honors including: IEEE Circuits and Systems Society Golden Jubilee Medal (1999), fellow of the IEEE (1990), Senior Research Fellowship of the Japan Society for Promotion of Science (1988), and Distinguished Professorship of the German National Science Foundation (1990). He has also held visiting positions at the Tokyo Institute of Technology and at the University of Karlsruhe, Germany. Dr. Thulasiraman has been vice-president of the IEEE Circuits and Systems Society (1998, 1999), technical program chair of the IEEE International Symposium on Circuits and Systems (1993, 1999), co-guest editor of a special issue on computational graph theory: algorithms and applications of the *IEEE Transactions on Circuits and Systems* (March 1988), associate editor of the *IEEE Transactions on Circuits and Systems* (1989-1991, 1999-2001) and founding regional editor of the *Journal of Circuits, Systems, and Computers* (since 1990).

Anindya Das received the MSc degree in mathematics from the Indian Institute of Technology, Kanpur, India, in 1982, the MS degree in computer science from the Indian Institute of Technology, Madras, India, in 1985, and the PhD degree in electrical and computer engineering from Concordia University, Montreal, Canada, in 1989. He is currently an assistant professor in the School of Computer Science, University of Oklahoma, Norman. His research interests include telecommunication network protocols, software engineering, network design and management, fault-tolerant computing, and distributed network optimization.