Information Sciences 203 (2012) 44–58

Contents lists available at SciVerse ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A study on the extended unique input/output sequence

Xinchang Zhang [a,c,*], Meihong Yang [a], Jian Zhang [b], Huiling Shi [a], Wei Zhang [a]

[a] Shandong Key Laboratory of Computer Networks, Shandong Computer Science Center, Jinan 250101, China
[b] Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
[c] Computer Network Information Center, Chinese Academy of Sciences, Beijing 100190, China

ARTICLE INFO

Article history:
Received 17 January 2011
Received in revised form 19 January 2012
Accepted 15 March 2012
Available online 23 March 2012

Keywords:
Protocol conformance testing
Unique input/output sequence
Finite state machine
State

ABSTRACT

The unique input/output (UIO) sequence is an important state identification technique in the FSM-based protocol conformance testing. However, some states of a FSM might have no UIO sequence. To address the above problem, this paper introduces an extended UIO sequence, called a GUIO sequence, which distinguishes a designated state from the remaining states by a group-by-group means. The problem of searching the optimal GUIO sequence is NP-hard. In this paper, we present a GUIO search method based on a greedy heuristics and a φ-hop search approach. The proposed method can obtain desirable GUIO sequences with relatively low computational complexity.

1. Introduction

Complex communication system employs a precise set of rules, called a protocol, to define the interactions among the elements of the system. A protocol implementation which is expected to be interoperable is developed in terms of the corresponding protocol specification. However, non-interoperability between protocol implementations may occur by some reasons, such as the ambiguity and/or misinterpretation of the protocol specifications and different implementations using different options in specifications. Therefore protocol conformance testing is an indispensable part before the implementation of a protocol is deployed in the practical networks.

Typically, the implementation of a protocol is tested for conformance by applying a sequence of inputs and verifying whether or not the expected sequence of outputs is produced. In some cases, the control flow of the protocol specification can be interpreted in the form of the finite state machine (FSM), e.g., [7,30,31,33]. Moreover, FSMs are the underlying models for some formal description techniques, such as SDL and UML State Diagrams. Therefore the problem of developing formal techniques for generating test sequences for FSMs has been an active research area in the past decades. For examples, [4,14,15,27] presented some surveys of the FSM-based test, [5,6,8,10,13,22,25,28,29] proposed some typical test sequence derivation methods.

Sabnani and Dahbura [25] proposed a test sequence generation method based on the UIO sequence. A UIO sequence for the state (of a FSM) is an input/output behavior that is not exhibited by any other state. The UIO sequence provides a natural and practical means for verifying that the protocol implementation is in a given expected state [26]. The length of the UIO sequence generated by the method in [25] is not optimized, which produces extra test load. Aho and Dahbura [3] presented a technique which combines the rural Chinese postman tour and UIO sequence, to optimize the length of test sequences. More improved methods for computing the UIO sequence can been seen in [1,11,16,18,21,23,34].

* Corresponding author at: Shandong Key Laboratory of Computer Networks, Shandong Computer Science Center, Jinan 250014, China. Tel.: +86 531 82605213; fax: +86 531 82605529.
E-mail address: xinczhang@hotmail.com (X. Zhang).

0020-0255/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2012.03.007

Unfortunately, some FSMs do not possess a UIO sequence for each state [2,25,28]. To the best of our knowledge, there have been only few studies which attempt to solve the above problem. For the state which does not possess a UIO sequence, Sabnani and Dahbura [25] used a signature to distinguish the state from the remaining states by a one-by-one way. However, the above signature sometimes is invalid. Additionally, the test effectiveness of the above signature is low because of the unoptimized length. Ahmad et al. [2] presented a guided heuristic algorithm to synthesize FSMs such that each state has a UIO sequence. In the algorithm, the transitions are augmented by adding extra outputs incrementally. In addition, the algorithm needs some special terminals which are not a part of the system under test. The protocol conformance testing is usually a black-box test, which means that it is difficult to add extra outputs to the implementation. Therefore the method in [2] has some practical limitations in the protocol conformance testing.

This paper uses a special sequence (called returned transfer sequence) to address the above invalid signature problem, and further proposes a method for generating an extended UIO sequence for the states that do not possess UIO sequences. Similar to [25], the proposed method makes the following two assumptions: (1) The FSM is minimal (called the minimality assumption); (2) The FSM can be moved from any state to its initial state by a reset input (called the reset assumption). Actually, most UIO-based test sequence generation methods also make the above two assumptions. The extended UIO sequence generated by the proposed method, called the GUIO sequence, distinguishes a designated state from the remaining states by a group-by-group way. Therefore the GUIO sequence is usually shorter than the corresponding sequence generated by the one-by-one way. To reduce the computational complexity, this paper uses a greedy heuristics and a φ-hop search approach to generate the GUIO sequence.

The rest of the paper is organized as follows. In Section 2, we introduce the related work. The finite state machine and UIO sequence are introduced in Section 3. Section 4 describes the two main problems of obtaining the extended UIO sequence. We introduce the GUIO sequence generation method in Section 5, and evaluate the method by analyzing the experimental results in Section 6. Finally, we conclude our work in Section 7.

2. Related work

In the UIO method [25], the test sequence is generated based on the unique input/output sequence (UIO). As mentioned previously, a UIO sequence for a designated state is an input/output behavior that is not exhibited by any other state. When the input part of the UIO sequence for a state (denoted by s) is applied to the FSM, the output sequence is compared with the expected output sequence. If they are the same, then the FSM is in the state s. Otherwise, the FSM is not in the state s. The time complexity of the UIO search algorithm in [25] is $O(n^2(d_{max})^{(2n^2+2)})$. Except the minimality and reset assumptions, the UIO method assumes that the machine remains in its current state and produces a *null* output when an unspecified input is applied to a certain state (called the completeness assumption). Many test sequence generation methods (i.e., W method [5] and Wp method [8]) also make the above three assumptions.

The UIO sequence and UIO-based test have been widely studied in the past decades, e.g., [1,11,12,16,18,20,32,34]. Ramalingam et al. [24] compared the lengths of the sequences generated by some typical test sequence generation methods for the conformance testing. Assume that: (1) The characterizing set for each state of the C method [13], W method [5] and Wp method [8] is the union of the UIO sequence selected for the UIO method; (2) The number of states of the implementation under test is no more than the number of states of the specification FSM. Then the order among the lengths of test sequences generated by the various methods is: $L(UIO) \leqslant L(Wp) \leqslant L(W)$ and $L(UIO) \leqslant L(C)$, where $L(X)$ represents the length of a test sequence selected in method X [24].

Vuong et al. [28] presented an improved UIO-based test sequence generation method (called UIOv method) to improve the fault coverage. In addition to the minimality, reset and completeness assumptions, the UIOv method assumes that both the specification and implementation FSMs are completely specified. The method checks whether the selected UIO sequences of the states of the specification FSM are also UIO sequences of the corresponding states in the implementation FSM. Another approach for improving the fault coverage of the UIO-based test can be seen in [20]. [26] presents a technique to reduce the execution time of test sequences generated by the method in [20]. In the technique, multiple UIO sequences are generated per state, an appropriate assignment of edges to UIO sequences decreases the number of transitions required in the resulting tour. Aho and Dahbura [3] proposed a technique to combine the rural Chinese postman tour and UIO sequence. Zhang and Probert [34] presented a mathematical model to construct a minimum-length test sequence in the case where each state possesses a UIO sequence.

Naik [21] studied the algorithms for computing all UIO sequences of minimal lengths, and computing the UIO sequence for each state if it exists. The key idea of the algorithms is that it is possible to compute UIO sequences for all possible states from a small set of initial UIOs in a large class of machines. Ahmad et al. [1] proposed an algorithm (called LANG) for computing UIO sequences for completely and incompletely specified FSMs. In the algorithm, the sets of states which produce identical outputs for non-conflicting inputs are partitioned by incrementally constructing a successor tree.

Searching UIO sequences based on evolutionary algorithms (EAs) have recently been researched (e.g., [16–19]). Lehre and Yao [16] presented a rigorous runtime analysis of the limitations of using the EA-based method for reducing the length of the UIO sequence. Lehre and Yao [17] investigated the choice of acceptance criterion in the (1 + 1) EA, and proved that changing these parameters can reduce the runtime from exponential to polynomial for some instance classes of the UIO problem. Li et al. [18] used the genetic algorithm to search the UIO sequence to reduce the length of the found UIO sequence. Li et al. [19]