

改进的错误诊断算法^①

崔 玲^{②***} 张建标^{③****} 郑 爽* 李秀云*

(* 北京工业大学信息学部 北京 100124)

(** 可信计算北京市重点实验室 北京 100124)

(*** 信息安全等级保护关键技术国家工程实验室 北京 100124)

摘 要 考虑到在检测到错误后需诊断错误,而目前的错误诊断算法多是基于单个错误的假设,且算法复杂度较高,在分析主动测试和被动测试中分别采用经典错误诊断算法——Ghedamsi 算法和 Miller 算法的基础上,提出了一种改进的错误诊断算法。该算法基于主动测试,先根据测试结果获得初始诊断集,然后由被动测试的逆向判定法得到错误诊断集,最后由测试序列进一步筛选得到最终的错误诊断结果。在生成和验证诊断集时,只对包含可疑转换的测试用例执行遍历操作,使算法更趋高效。算法分析和实例验证结果表明,该算法能够高效地诊断单个错误。

关键词 有限状态机(FSM),一致性测试,主动测试,错误诊断

0 引 言

基于有限状态机(finite state machine, FSM)的一致性测试问题得到了广泛的研究。一致性测试旨在检验所实现的协议实体(或系统)与协议规范的符合程度,即测试一个协议给定实现的外部行为是否符合协议的规范。一致性测试的方法有主动测试和被动测试两类,主动测试是主动设计一些测试序列来测试,而被动测试则是通过分析监听采集的数据来发现被测系统中存在的错误^[1],它不影响系统的正常工作。如果实际输出与期望输出不一致,则表示被测系统不符合设计规范,存在错误。如何诊断或定位错误是一个比较复杂的问题。

主动测试中的经典错误诊断算法是Ghedamsi提出的在单个错误的前提下的算法^[2],该算法的基本思想是枚举所有可能的错误情形并设计相应的判定实验来排除其中一些候选错误。赵保华^[3]对Ghe-

damsi 算法进行了改进,利用可疑转换的下一个输入/输出对的头状态集合等信息快速地排除一些候选错误,降低了时间复杂度。贾家涛^[4]提出了一种新的问题模型,并结合程序错误定位技术,提出了一种改进算法。在被动测试中,Miller^[5]提出的算法较为经典,他利用逆向判定法给出了诊断单个错误的方法,时间复杂度较低,不过一些重复错误(在一条测试序列中多次出现的错误)不能被检测。Guo^[6]给出了两个针对 Miller 算法的改进算法,能进一步缩小错误诊断范围。唐启锋等人^[7]提出了一种综合多种错误诊断方法的错误定位模型,将多种已有错误定位方法所输出的结果进行组合以获得更好的错误定位效果。Lei^[8]指出软件测试和错误定位技术的有效互动可以极大改善错误定位效率。另外,将统计学、数据挖掘等理论应用于错误定位技术中有着广泛的研究前景^[9]。

本文分析了 Ghedamsi 算法和 Miller 算法这两种经典算法,提出了一种改进的错误诊断算法,与这

① 国家自然科学基金(61501007)资助项目。
② 女,1979 年生,讲师;研究方向:信息安全;E-mail: vicky@bjut.edu.cn
③ 通信作者,E-mail: zjb@bjut.edu.cn
(收稿日期:2017-07-07)

两种方法相比,本文算法更清晰高效,能准确地诊断单个错误。

1 术语定义

定义 1 FSM: 一个 FSM 模型 M 是一个六元组: $M = (S, I, O, q_0, \delta, \lambda)^{[8]}$, 其中 S 是有穷的状态集合, I 是有穷的输入集合, O 是有穷的输出集合, s_0 是 FSM 模型的初始状态, 且 $s_0 \in S$; $\delta: S \times I \rightarrow S$ 为状态转换函数, $\lambda: S \times I \rightarrow O$ 为输出函数。

用 $t_{ij}: s_i \xrightarrow{x/y} s_j$ 表示一个转换, 其中 $s_i \in S$, $s_j \in S$, $x \in I^+$, $y \in O^+$, $\delta(s_i, x) = s_j$, $\lambda(s_i, x) = y$, 其含义可描述为当 FSM 模型处于 s_i 状态时, 对其输入序列 x , FSM 模型将转入状态 s_j , 同时输出序列 y 。

诊断多个错误问题的算法复杂度是指数级的^[10], 因此多数文献提出的错误诊断算法都是假定只存在单个输出错误或单个转换错误。本文提出的算法同样基于这种假设。

假设已有测试序列集 $TC = \{tc_1, tc_2, \dots, tc_n\}$, 其中每个 tc_m 都是一个测试序列, 对应输入序列 $i_m = \{i_{m1}, i_{m2}, \dots, i_{mp}\}$, 预期输出序列 $o_m = \{o_{m1}, o_{m2}, \dots, o_{mp}\}$, 实际输出序列 $o'_m = \{o'_{m1}, o'_{m2}, \dots, o'_{mp}\}$ 。

定义 2 症状: 若在 o_i 和 o'_i 中有 $o_{ij} \neq o'_{ij}$, 则称其为症状 (symptom)。

定义 3 初始症状: 若在 o_i 和 o'_i 中有 $o_{ij} \neq o'_{ij}$, 且对任意 $1 \leq k \leq j$ 有 $o_{ik} = o'_{ik}$, 则称 $o_{ij} \neq o'_{ij}$ 为测试序列 tc_i 的初始症状。

定义 4 可疑转换: 与症状对应的转换称为可疑转换 (symptom transition)。

定义 5 唯一可疑转换: 如果所有症状对应同一个可疑转换, 则称该可疑转换为唯一可疑转换 (unique symptom transition, ust), 由唯一可疑转换产生的输出叫做唯一症状输出 (unique symptom output, uso)。

定义 6 冲突集: 在某个症状 e 之前测试序列所经过的转换称为该症状的冲突集, 记为 $CS(e)$ 。

定义 7 前向轨迹: 称状态集序列 $L^0, L^1, \dots, L^n$

为前向轨迹。其中, L^0 为系统所有状态集合, L^{i+1} 是由所有输入输出为 i_j/o_j 所对应的转换的头状态在 L^i 中的尾状态构成的。

定义 8 后向轨迹: 称状态集序列 $(L^k)^R, (L^{k-1})^R, \dots, (L^0)^R$ 为后向轨迹。其中, $(L^k)^R$ 为系统所有状态集合, $(L^{k-1})^R$ 是由所有输入输出为 i_k/o_k 所对应的转换的尾状态在 $(L^k)^R$ 中的头状态构成的。

2 已有算法

Ghedamsi 算法是主动测试中的经典错误诊断算法, 主要步骤如下:

第 1 步: 生成所有的症状。

第 2 步: 为每个症状产生冲突集。在只存在单个错误的假设下, 错误肯定发生在症状之前。取这些冲突集的交集作为初始候选 (initial tentative candidate, ITC) 集。将 ITC 分为两个非交子集合, 一个称为 ustset, 仅包含唯一可疑转换 (ust), 另外一个子集则称为最终候选 (final tentative candidate, FTC) 集。

第 3 步: 分别处理 ust 和 FTC。如果存在 ust, 则对 ust 的处理如下, 遍历所有测试序列, 如果转换是 ust, 则检查实际输出是否是 uso, 如果所有的实际输出均为唯一症状输出 (uso), 则可以确定 ust 是错误转换。对 FTC 的处理如下, 为 FTC 中每个转换 t_k 枚举所有可能的转换错误, 然后遍历所有测试用例, 检验引入该错误是否能产生所有的实际输出, 若回答肯定则记录该转换错误对应的尾状态, 每一个转换的错误尾状态集合都对应着可能存在的错误转换。

第 4 步: 再设计一些测试用例进一步筛选由上面 3 步得到的错误集合。

赵保华主要在两个步骤中对 Ghedamsi 算法进行了改进。首先在第 1 步, 生成的不是所有的症状, 而是初始症状, 作者已证明由所有的初始症状产生的 ITC 与由所有症状产生的 ITC 是相同的^[11]。其次是第 3 步处理 FTC 的过程, 利用可疑转换的下一个输入/输出对的头状态集合来刷新该转换的错误

尾状态集合,有效地降低了算法复杂度。

Miller 的逆向判定法是基于被动测试的经典算法。被动测试与主动测试一个显著的不同就是被动测试无法获取被测系统的当前状态。根据观察到的输入/输出序列,并利用状态机的转换信息来推断被测系统在某个时刻所处的状态集合,若该集合在某个状态为空,则认为被测系统存在错误。该算法主要步骤如下:

- (1) 构造前向轨迹 $L^0, L^1, \cdots, L^n$ 。如果有 $L^{k-1} \neq \emptyset, L^k = \emptyset$, 则表明在第 k 步检测到错误。
- (2) 构造后向轨迹 $(L^k)^R, (L^{k-1})^R, \cdots, (L^0)^R$ 。Miller 证明必定存在 j , 使得 $L^j \cap (L^j)^R = \emptyset$ 。
- (3) 根据如下两个定理推断可能存在的错误。

定理 1 如果 L^j 中有一个状态 s_p , 在输入 i_{j+1} 下的输入不等于期望输出, 并且在 $(L^{j+1})^R$ 中有 $\delta(s_p, i_{j+1})$ 这个元素, 则可能存在输出错误 $s_p \xrightarrow{i_{j+1}/o_{j+1}} \delta(s_p, i_{j+1})$ 。

定理 2 如果 L^j 中有一个状态 s_p , 存在转换 $s_p \xrightarrow{i_{j+1}/o_{j+1}} s_q$, 并且有状态 $s_r \in (L^{j+1})^R$, 则可能存在转换错误 $s_p \xrightarrow{i_{j+1}/o_{j+1}} s_r$ 。

利用一个指数时间算法可以指出 Miller 算法不能诊断一些重复错误^[12]。这是该算法的局限,原因是被动测试中测试序列是实际运行中的输入数据,输入值和序列长度均不可选择。因此本文考虑将 Miller 算法应用到主动测试中,既保留该算法较低的时间复杂度,又可利用主动测试可选择的测试序列进行筛选以期获得最小的错误诊断集。

3 本文算法

本文算法包括 4 个步骤,其中前两步采用赵保华的改进算法,第 3 步处理 FTC 时与之不同,本文将包含 FTC 的测试序列作为 Miller 算法的测试序列,利用该算法求得最终的错误诊断集。下面结合实例,给出算法的具体步骤。

图 1 (a) 是一个协议描述的有限状态机^[2],图 1 (b) 是实现后的有限状态机,其中转换 $s_2 \xrightarrow{a/0} s_3$ 被错误地实现为 $s_2 \xrightarrow{a/0} s_5$ (在图中以虚线标出)。

根据唯一输入输出(unique input output, UIO)方法生成测试序列集 $TC = \{raab, rbaa, raaaa, rabba, rbaba, rbbab, rbaaab, rbabaa, rbbaaa, rbbbba, rbabaaa, rbabbbba\}$,每个状态的 UIO 序列如表 1 所示。测试时各个测试序列的预期输出和实际输出如表 2 所示。

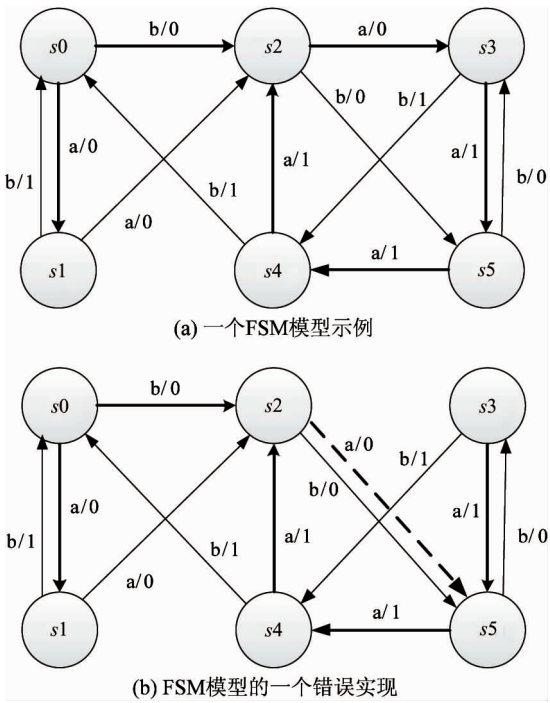


图 1 一个示例 FSM 模型及其实现

表 1 示例有限状态机的 UIO 序列

状态	UIO 序列	状态	UIO 序列
s0	b/0 a/0	s3	b/1 a/1
s1	a/0 b/0	s4	a/1 a/0
s2	a/0 a/1	s5	a/1 b/1

表 2 预期输出和实际输出

序列号	测试序列	预期输出	实际输出
tc1	raab	000	000
tc2	rbaa	001	001
tc3	raaaa	0001	0001
tc4	rabba	0100	0100
tc5	rbaba	0011	00 <u>0</u> 1
tc6	rbbab	0011	0011
tc7	rbaaab	00111	0011 <u>0</u>
tc8	rbabaa	00110	00 <u>0</u> 1 <u>1</u>
tc9	rbbaaa	00110	00110

(续表 2)

tc10	rbbbbba	00011	00011
tc11	rbabaaa	001101	00 <u>0</u> 1 <u>1</u>
tc12	rbabbba	001100	00 <u>0</u> 1 <u>1</u> 0

3.1 生成所有初始症状

根据测试序列的预期输出和实际输出生成所有初始症状 $Symp_i$ 。示例中 tc5、tc8、tc11 和 tc12 的初始症状相同,对应的可疑转换均为 $s3 \xrightarrow{b/1} s4$, tc7 的初始症状对应的可疑转换为 $s4 \xrightarrow{b/1} s0$ 。

3.2 求初始候选集 ITC

为每一个初始症状产生冲突集,并求得其交集初始候选(ITC)。示例中,tc5、tc8、tc11 和 tc12 中初始症状的冲突集均为 $\{s0 \xrightarrow{b/0} s2, s2 \xrightarrow{a/0} s3, s3 \xrightarrow{b/1} s4\}$, 对应测试序列段为 bab; tc7 中初始症状的冲突集为 $\{s0 \xrightarrow{b/0} s2, s2 \xrightarrow{a/0} s3, s3 \xrightarrow{a/1} s5, s5 \xrightarrow{a/1} s4, s4 \xrightarrow{b/1} s0\}$, 对应测试序列段为 baaab。取这些冲突集的交集得到 $ITC = \{s0 \xrightarrow{b/0} s2, s2 \xrightarrow{a/0} s3\}$ 。所有冲突集对应的测试序列段为 $\{bab, baaab\}$ 。

3.3 产生错误诊断集 ErrorSet

首先处理 ust。

算法 1: 处理 ust

输入:ust;

输出:ErrorSet;

Procedure ust-processing(ust)

```

1  For all  $tc_i \in TC$  Do
2  For all  $I_{ij} \in tc_i$  Do
3  If ( $t_{ij} = ust$ ) Then
4  If ( $o'_{ij} \neq uso$  or  $o'_{ij} \neq o_{ij}$ ) Then
5      {ustset =  $\emptyset$ ; exit}
6  End
7  End
8  ErrorSet = ustset
```

经过算法 1 处理 ust,最后 ErrorSet 若非空,则表示 ust 是可能的输出错误,在单个错误的假设下,诊断结束。

图 1 例子中不存在唯一可疑转换 ust,所以 $FTC = ITC$ 。

然后处理 FTC,具体过程如下:

(1) 选取测试序列。遍历所有冲突集对应的测试序列段,记录每个测试序列段的出现次数,从中选择出现次数最多的测试序列段,记为 tc_{Miller} 。观察 tc_{Miller} 及对应实际输出,组成测试输入/输出序列,作为 Miller 算法的测试序列。

示例中 $tc_{Miller} = \{bab\}$, 其对应的实际输出均为 000,组成输入/输出序列 $\{b/0, a/0, b/0\}$ 。

(2) 构造前向轨迹。主动测试可以确定 $L^0 = \{s0\}$, 根据定义 7 得到: $L^1 = \{s2\}$, $L^2 = \{s3\}$, $L^3 = \{\emptyset\}$, 错误发生在第三步。

(3) 构造后向轨迹。根据定义 8 求得后向轨迹: $(L^3)^R = \{s0, s1, s2, s3, s4, s5\}$, $(L^2)^R = \{s0, s2, s5\}$, $(L^1)^R = \{s1\}$, $(L^0)^R = \{\emptyset\}$ 。其中, $L^1 \cap (L^1)^R = \{\emptyset\}$, $L^2 \cap (L^2)^R = \{\emptyset\}$, $L^3 \cap (L^3)^R = \{\emptyset\}$ 。

(4) 推断可能的错误。找出 $L^j \cap (L^j)^R = \emptyset$ 的情况,根据定理 1 和定理 2 推断可能存在的错误。将这些错误加入到错误诊断集 ErrorSet 中。

L^0 中有状态 $s0$, 对于 i_1/o_1 即 $b/0$ 存在转换 $s0 \xrightarrow{b/0} s2$, 且 $(L^1)^R$ 中有状态 $s1$, 因此可能有尾状态错误 $s0 \xrightarrow{b/0} s1$; L^1 中有状态 $s2$, 转换 $s2 \xrightarrow{a/0} s3$, 且 $(L^2)^R$ 中有 $s0, s2, s5$, 因此可能有 3 个尾状态错误: $s2 \xrightarrow{a/0} s0, s2 \xrightarrow{a/0} s2, s2 \xrightarrow{a/0} s5$; L^2 中有状态 $s3$, 转换 $s3 \xrightarrow{b/1} s4$, 且 $(L^3)^R$ 有 $s4$, 因此可能有输出错误 $s3 \xrightarrow{b/0} s4$ 。最后得出错误诊断集 $ErrorSet = \{s0 \xrightarrow{b/0} s1, s2 \xrightarrow{a/0} s0, s2 \xrightarrow{a/0} s2, s2 \xrightarrow{a/0} s5, s3 \xrightarrow{b/0} s4\}$ 。

算法 2: 处理 FTC

输入:FTC;

输出:ErrorSet;

Procedure FTC-processing(FTC)

```

1   $tc_{Miller} = tc_{Csi} \mid \text{the times of } tc_{Csi} \text{ is max}$ 
2   $L^0 = \{s0\}$ 
3  For  $j$  from 1 to  $m$  Do / *  $m = \mid tc_{Miller} \mid$  */
4  If ( $L^{j-1} \neq \emptyset$ ) Then
```

```
5   $L^j = \{s \mid s \in S \& s_p \in L^{j-1} \& \delta(s_p, i_j) = s \& \lambda(s_p, i_j)$   
    $= o_j\}$   
6  End  
7   $(L^m)^R = S$   
8  For  $j$  from  $m - 1$  to 0 Do  
9   $(L^j)^R = \{s \mid s \in S \& s_q \in (L^{j+1})^R \& s_q = \delta(s, i_j)$   
    $\& \lambda(s, i_j) = o_j\}$   
10 End  
11 For  $j$  from 0 to  $m$  Do  
12 If  $(L^j \cap (L^j)^R = \emptyset)$  Then  
13 For all  $s_p \in L^j$  Do  
14 If  $(\lambda(s_p, i_{j+1}) \neq o_{j+1} \text{ and } \delta(s, i_{j+1}) \in (L^{j+1})^R)$   
   Then  
15  $ErrorSet = ErrorSet \cup s_p \xrightarrow{i_{j+1}/o_{j+1}}$   
    $\delta(s_p, i_{j+1})$   
16 If  $(\lambda(s_p, i_{j+1}) = o_{j+1})$  Then  
17 For all  $s_r \in (L^{j+1})^R$  Do  
18  $ErrorSet = ErrorSet \cup \{s_p \xrightarrow{i_{j+1}/o_{j+1}} s_r\}$   
19 End  
20 End  
21 End
```

3.4 进一步筛选错误诊断集 ErrorSet

对于 ErrorSet 中每一个元素,遍历所有产生初始症状的测试序列,比较期望输出与实际输出是否一致,如果不一致则将该元素从 ErrorSet 中删除。

对于 $s0 \xrightarrow{b/0} s1$, 有 tc11:rbabaaa, 期望输出 000110, 与实际输出 000111 不一致, 从 ErrorSet 中去掉 $s0 \xrightarrow{b/0} s1$; 对于 $s2 \xrightarrow{a/0} s0$, 有 tc5:rbaba, 期望输出 0010, 与实际输出 0001 不一致, 从 ErrorSet 中去掉 $s2 \xrightarrow{a/0} s0$; 对于 $s2 \xrightarrow{a/0} s2$, 有 tc11:rbabaaa, 期望输出

000110, 与实际输出 000111 不一致, 从 ErrorSet 中去掉 $s2 \xrightarrow{a/0} s2$; 对于 $s2 \xrightarrow{a/0} s5$, 没有不一致的情况; 对于 $s3 \xrightarrow{b/0} s4$, 有 tc8:rbabaa, 期望输出 00010, 与实际输出 00011 不一致, 从 ErrorSet 中去掉 $s3 \xrightarrow{b/0} s4$ 。因此最后 $ErrorSet = \{s2 \xrightarrow{a/0} s5\}$, 与 FTC 比较, 可知错误原因是 $s2 \xrightarrow{a/0} s3$ 被错误地实现为 $s2 \xrightarrow{a/0} s5$, 诊断结束。

4 算法分析和比较

假定有限状态机(FSM)的状态数为 n , 测试序列集包含 Ls 个测试用例, 最长的测试用长度为 Lc 。算法的第 1 步、第 2 步和算法 1 只需遍历一次测试序列集, 时间复杂度为 $O(LsLc)$ 。算法 2 处理 FTC 步骤中, 第 1 行需要比较所有冲突集对应的测试序列段, 最大比较次数为 $LsLc$, 因此时间复杂度为 $O(LsLc)$ 。第 3 行至第 21 行是 Miller 的逆向判定法, 其时间复杂度为 $O(n^2LsLc)$ 。算法第 4 步主要是对于每一个可能错误, 遍历所有产生初始症状的测试序列, 时间复杂度最大为 $O(n^2LsLc)$ 。因此文本算法的时间复杂度最大为 $O(n^2LsLc)$ 。

表 3 给出了本文算法和已有算法时间复杂度的比较, 表 4 给出了本文算法与这些算法具体的不同之处。赵保华的改进算法不能保证可获得最小的错误诊断集, 因此还需要再设计一些测试用例进行筛选^[12]。而本文算法在算法复杂度分析中已包含测试用例的筛选步骤。综合分析, 本文算法能够高效地诊断单个错误。

表 3 算法复杂度比较

	本文算法	Ghedamsi 算法	赵保华算法	Miller 算法
时间复杂度	$O(n^2LsLc)$	$O(nLsL^3c)$	$O(nLsL^2c)$	$O(n^2LsLc)$

表 4 算法内容比较

	比较点	本文改进之处
Ghedamsi 算法	1. 产生所有症状 2. 枚举所有可能错误,然后进行验证	1. 产生初始症状 2. 根据前向轨迹和后向轨迹提供的信息快速缩小诊断范围
赵保华的算法	1. 根据可疑转换的下一个输入/输出对的头状态集合信息缩小错误诊断集 2. 缩小和验证错误诊断集时遍历的是所有测试序列	1. 根据 Miller 算法直接生成错误诊断集,时间复杂度低 2. 生成和验证错误诊断集时遍历的是产生初始症状的测试序列
Miller 算法	1. 基于被动测试,测试用例不可选,不能诊断重复错误	1. 基于主动测试,测试用例可选且可增加,可以诊断重复错误

5 结 论

本文分析了一致性测试的主动测试和被动测试两种经典算法,结合其优点,提出了一种改进算法。该算法先根据测试结果获得初始诊断集,然后由被动测试的逆向判定法得到错误诊断集,最后由测试序列进一步筛选得到最终的错误诊断结果。并且在生成和验证诊断集时,只对包含可疑转换的测试用例执行遍历操作,使算法更趋高效。通过算法分析和实例可以看出,该算法简单有效,能准确地诊断出单个错误。

不过本文算法仍旧是基于单个错误的假设,在后续研究中,将考虑多个错误的诊断,以及在多种状态机模型上的应用。

参考文献

[1] Lee D, Yannakakis M. Principles and methods of testing finite state machines——a survey [J]. *Proceedings of IEEE*, 1996,84:1090-1126

[2] Ghedamsi A, Von Bochmann G. Test result analysis and diagnostics for finite state machines[C]. In: *Proceedings of the 12 th International Conference on Distributed Computing Systems*, Yokohama, Japan, 1992. 244-251

[3] 赵保华,钱兰,周颢,等. 基于有限状态机的错误诊断算法[J]. *电子与信息学报*, 2006,28(9):1679-1683

[4] 贾家涛,曲明成,吴翔虎. 一种改进的通信有限状态机的错误诊断方法[J]. *智能计算机与应用*, 2016,6(4):

51-54

[5] Miller R E, Arisha K A. Fault identification in networks by passive testing[C]. In: *Proceedings of the 34th Annual Simulation Symposium*, Seattle, USA, 2001. 277-284

[6] Guo X H, Zhao B H, Qian L. Fault identification by passive testing[C]. In: *Proceedings of International Conference on Telecommunications*, Fortaleza, Brazil, 2004. 826-834

[7] 唐启锋,许蕾,钱巨,等. 一种基于模型结合的错误定位方法[J]. *中国科技论文*, 2012,7(1): 19-27

[8] Lei Y, Mao X G, Dai Z Y, et al. Effective fault localization approach using feedback[J]. *IEICE Transactions on Information and Systems*, 2012, 95(9):2247-2257

[9] 曹鹤玲,姜淑娟,鞠小林. 软件错误定位研究综述[J]. *计算机科学*, 2014,41(2):1-6,14

[10] Belhassine cherif R, Ghedamsi A. Multiple fault diagnostics for communicating nondeterministic finite state machines[C]. In: *Proceedings of the 6th IEEE Symposium on Computers and Communications*, Hammamet, Tunisia, 2001. 661-666

[11] 赵保华,钱兰,周颢,等. 基于 UIO 测试序列的错误诊断算法[J]. *电子与信息学报*, 2006, 28(11):2152-2156

[12] 钱兰. 协议一致性测试的错误诊断研究:[博士学位论文][D]. 合肥:中国科学技术大学计算机科学技术系, 2006. 67-68

An improved fault diagnosis algorithm

Cui Ling^{***}, Zhang Jianbiao^{*** **}, Zheng Shuang^{*}, Li Xiuyun^{*}

(^{*} Department of Information, Beijing University of Technology, Beijing 100124)

(^{**} Beijing Key Laboratory of Trusted Computing, Beijing 100124)

(^{***} National Engineering Laboratory for Critical Technologies of Information
Security Classified Protection, Beijing 100124)

Abstract

The active test in conformance testing is studied. Considering that after a fault is detected, the fault diagnosis should be performed, while most of the current fault diagnosis algorithms are based on the assumption of single fault and have the higher complexity, an improved fault diagnosis algorithm is proposed based on the analysis of the two classical algorithms, i. e. Ghedamsi algorithm for active test and Miller algorithm for passive test. This proposed algorithm obtains the initial diagnosis set according to test results under the active test, and then uses the reverse determination method for passive testing to obtain the fault diagnosis set. Finally, the test sequence further screens the diagnosis set. The algorithm analysis and example verification prove that the algorithm can diagnose single fault efficiently.

Key words: finite state machine (FSM), conformance testing, active test, fault diagnosis