基于通信协议的接口测试用例自动生成框架

刘 逻，郭立红，陈 媛，王俊杰
（中国科学院长春光学精密机械与物理研究所，吉林 长春 130033）

摘要： 为了实现对软件配置项和软件系统的接口测试的测试用例自动生成，建立了基于通信协议的接口测试用例自动生成框架，对该生成框架的输入模型和算法集合进行研究。提出了基于通信协议的接口测试用例生成框架的输入模型，并在输入模型中建立数据帧之间的一级关联矩阵和数据帧内字段间的二级关联矩阵。以输入模型为基础，说明发送端和接收端的接口测试用例自动生成算法。结合实际项目中的通信协议，介绍了如何将通信协议信息转换至输入模型，如何通过算法集合自动生成测试用例集。实验结果表明：基于通信协议的接口测试用例自动生成框架生成的接口测试用例可以替代人工生成的接口测试用例，极大地提高了软件接口测试工作的效率。

关键词： 接口测试；自动生成框架；输入模型；关联矩阵；生成算法
中图分类号： TP311.52 **文献标识码：** A **文章编号：** 1000－8829(2020)01－0046－09
doi： 10.19708/j.ckjs.2020.01.009

Automated Test Case Generation Framework Based on Interface Communication Protocol

LIU Luo, GUO Li-hong, CHEN Yuan, WANG Jun-jie
(Changchun Institute of Optics, Fine Mechanics and physics, Chinese Academy of Sciences, Changchun 130033, China)

Abstract: In order to realize automatic test case generation for the interface testing of software configuration item and software system, an automatic test case generation framework based on interface communication protocol is established, and the input model and algorithm set of the generation framework are investigated. The input model of the interface test generation framework based on communication protocol was proposed, and the first-level correlation matrix between data frames and the second-level correlation matrix between fields in the data frame were constructed. An automatic generation algorithm of interface test cases containing sender and receiver was described based on the input model. Combined with the communication protocol of actual project, how to convert communication protocol information to the input model and how to automatically generate the test case set through the algorithm were introduced. The experimental results indicate the interface test cases generated by the proposed automatic test case generation framework based on interface communication protocol can replace the artificially generated interface test cases, which greatly improves the efficiency of software interface testing.

Key words: interface testing; automatic generation framework; input model; correlation matrix; generation algorithm

收稿日期： 2019－07－24
基金项目： 中国科学院仪器设备功能开发技术创新项目(Y9A034S)
作者简介： 刘逻(1983—)，男，博士，副研究员，主要研究方向为软件可靠性和软件测试；[通信作者] 郭立红(1964—)，女，博士，研究员，主要研究方向为计算机应用。

引用格式： 刘逻，郭立红，陈媛，等. 基于通信协议的接口测试用例自动生成框架[J]. 测控技术，2020，39(1)：46－54.
LIU L, GUO L H, CHEN Y, et al. Automated Test Case Generation Framework Based on Interface Communication Protocol[J]. Measurement & Control Technology, 2020, 39(1): 46－54.

软件测试是一项耗时且开销巨大的工作，有研究表明测试开销可达整个软件开发过程开销的 40% 以上。软件的配置项数量一般较多，少则几个，多则几十个，因此，在软件配置项测试和系统软件测试阶段需要投入的人力和时间开销更是巨大。其中，对于软件配置项和系统测试来说，功能测试和接口测试占据了软件测试总测试用例数的 90% 以上，而功能测试和接口测试在配置项测试和系统测试中存在众多交叉，接口测试的测试用例数常占测试用例总量的 50% 甚至更高。因此，若能建立接口测试用例自动生成方法，可以为软件配置项和系统测试节约大量的人力和时间。

当前，直接针对通信协议进行测试用例自动生成的技术较少。对于测试用例自动生成技术来说，主要分成面向路径的测试用例自动生成和面向需求的测试用例自动生成。面向路径的测试用例生成问题是一类约束满足问题，这类问题需要通过合理的搜索或寻优算法来求解[1]。但该方法需要根据软件的需求绘制软件系统的控制流和数据流图，然后再进行基于路径的测试用例自动生成，要解决路径的搜索空间组合爆炸问题，需要耗费大量的时间和人工，并且需要通过人工的方式对测试用例进行一轮筛选，剔除掉无效的测试用例，不太符合实际的被测软件系统，特别是任务周期紧的项目。

Chawla[2] 等人提出了基于粒子群和遗传算法的自动测试数据生成框架，但该框架不适用于配置项阶段的接口测试用例生成。Sofokleous[3]、姜淑娟[4] 等人提出的方法，均可提高路径覆盖测试效率，但用例覆盖率的程度依赖于控制流图和数据流图的质量。杨瑞[5] 提出了一种基于 EFSM(Enhanced Finite State Machine，扩展有限状态机) 模型的自动化测试用例生成方法，在此基础上提出了适应度函数(Fitness Function) ，并利用分散搜索(Scatter Search) 算法在排序后的候选路径集中挑选可行路径并自动生成相应的测试用例，该方法也是面向路径的测试用例自动生成方法，在系统庞大且通信接口众多的情况下，需要解决路径的搜索空间组合爆炸问题。丁蕊[6] 等人提出了关键点路径表达式能够计算程序的理论路径数目，且分支关键点成对出现的特点能够减少插桩工作量，但对于软件配置项和软件系统这类庞大的软件时，使用关键点路径的快速测试用例生成方法会遗漏非关键点路径。陈鑫[7] 等人提出了一种扩展 UML 活动图为列车控制系统中安全攸关场景建模的方法，设计了一种名叫 FPCC (FB-Path Complete Condition Test Coverage) 的结构测试覆盖组合，FPCC 的核心思想就是在每条敏感路径上将每个功能模块的路径进行全覆盖。Meiliana[8] 等人提出了基于 UML 活动图和时序图进行深度优先搜索来自动生成测试用例[8]。Samuel[9] 等人则在 UML 通信图的基础上建立簇级测试自动生成。Wu[10] 等人建立了 FPCC 方法，基于功能块图完成测试用例生成，而配置项或系统测试人员无法获得详细的功能块图作为测试依据。Gutierrez[11] 等人提出了一种基于模型驱动范式的系统过程来实现自动生成基于功能需求的功能测试用例，但使用该模型的需求需要按照面向对象的对应关系进行编写。聂长海[12] 等人提出了一种基于接口参数的黑箱测试用例自动生成算法，给出了对系统各个接口参数两两组合全面覆盖的原则来选择测试用例的方法，这种方法充分考虑了待测试软件的所有外部接口中的两个接口参数两两组合，但该方法对于接口中的指令类参数、状态类参数和数据类参数等数据类型没有进行分类处理，没有针对不同类型的参数进行特定的生成策略，没有针对接口是输入还是输出进行细分，因此在生成的测试用例上没有达到最优，并且对于混合型的通信协议不能直接使用。姚香娟[13] 等人提出一种基于神经网络的路径覆盖测试数据进化生成方法，利用训练好的神经网络粗略计算个体的适应值，但由于项目不同，因此训练集不能通用，对于不同的项目具有一定限制。

因此，从软件配置项测试和系统测试出发，利用各配置项之间的通信协议或被测系统与外部系统的通信协议，利用信息抓取和文本清洗技术将通信协议中的字段及信息进行提取。同时，根据设计出的基于通信协议的接口测试用例生成框架，将从通信协议中提取出的信息输入至三元组模型中的第一个元素，并由测试人员确定各数据帧之间是否具有一级关联性，数据帧内的字段是否具有二级关联性。然后，根据该三元组的第二个元素，接口测试用例自动生成算法将输入模型中的信息转换为接口测试用例，最后将生成的测试用例汇总至该三元组的第 3 个元素及测试用例集合中。以地面项目的通信协议作为实例，进行接口测试用例的自动生成，并通过人工生成的测试用例，参考文献 [12] 产生的测试用例和本文提出的方法进行比较。

1 基于通信协议的接口测试用例生成框架

接口有多种类型，常用的有 RS232 /422 /485、MIL-STD-1553B、CAN、Space Wire，以及以太网等。假设一个软件配置项拥有 N 个外部接口，因接口具有单向或双向性，所以当一个配置项的通信协议表单 $n \geqslant N$ 时，则每一张通信协议表单内包含了该接口通信的所有信息。结合自身实际软件测试工作，给出以下定义。

定义 1. 基于通信协议的接口测试用例生成框架，可标记为 TCGFICP(Test Case Generation Framework based on Interface Communication Protocol) ，并可形式

化地定义为三元组 TCGFICP = < IM,ALG,CASES >,其中,IM 表示基于通信协议的接口测试用例输入模型,ALG 表示基于通信协议的接口测试用例生成算法集合,CASES 表示生成的测试用例集合。

1.1　基于通信协议的接口测试用例输入模型的建立

定义 2. IM = < JK,SJZ,XQ > 。其中,JK 为接口描述规约集; SJZ 为接口通信协议内的数据帧规约集; XQ 为数据帧的详细描述规约集。

定义 3. JK = < JKMS,JKSX >,其中,JKMS 表示接口描述,描述该接口的发送和接收对象; JKSX 表示接口属性,描述该接口是发送端还是接收端。

对 JK 的各子元素定义如下:

定义 4. JKMS 表示通信协议中所有表单的集合,JKMS = { JKMS1,JKMS2,…,JKMSi,…,JKMSm},其中,JKMSi 表示通信协议中的第 i 张表单,m 表示通信协议中的表单总和。

定义 5. JKSX 表示通信协议中所有表单的属性的集合,JKSX = { RECI/SEND},其中,SEND 代表该接口为发送端,RECI 代表该接口为接收端。

定义 6. SJZ = < ZH,FM,FS,GLZH >,其中,ZH 表示帧号,描述该数据帧为该通信协议表单的第几个数据帧; FM 表示帧描述及该数据帧的意义; FS 表示帧属性; GLZH 表示该数据帧与其具有关联的数据帧的帧号的集合,若没有相互关联的数据帧,则置 0。

对 SJZ 的各子元素定义如下:

定义 7. ZH 表示帧号,ZH = < ZH11,ZH12,…,ZH1j,…,ZH21,…,ZHmn >,其中,ZH11 表示第 1 张通信协议表单的第一个数据帧,同理,依此类推,ZH1j 表示第 1 张通信协议表单的第 j 个数据帧,ZHmn 表示第 m 张通信协议表单的第 n 个数据帧。

定义 8. FM 表示数据帧的描述,FM 和 ZH 是一一对应的,FM = < FM11,FM12,…,FM1j,…,FM21,…,FMmn >,其中,FM11 表示第 1 张通信协议表单的第 1 个数据帧的描述内容,同理,依此类推,FM1j 表示第 1 张通信协议表单的第 j 个数据帧的描述内容,ZHmn 表示第 m 张通信协议表单的第 n 个数据帧的描述内容。

定义 9. FS 表示该帧的属性,FS = < COMMAND/STATE/DATA >,数据帧在通信过程中,会分为三类数据帧: 指令类、状态类和数据类。

定义 10. GLZH 表示该通信表单中数据帧之间的关联,矩阵 $\boldsymbol{GLY}m$ 为 $n \times n$ 的上三角矩阵,简称为一级关联矩阵。

$$\boldsymbol{GLY}m=\begin{bmatrix} GL_{m1} \\ GL_{m2} \\ GL_{m3} \\ \vdots \\ GL_{mn} \end{bmatrix}=\begin{bmatrix} 0 & GLY_{m12} & GLY_{m13} & \cdots & GLY_{m1n} \\ 0 & 0 & GLY_{m23} & \cdots & GLY_{m2n} \\ 0 & 0 & 0 & \cdots & \vdots \\ 0 & 0 & 0 & 0 & GLY_{m(n-1)n} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

其中,GLY_{m12} 表示第 m 张通信协议表单中,第 1 个数据帧和第 2 个数据帧之间的关系,同理,$GLY_{m(n-1)n}$ 表示第 m 张表单中,第 $n-1$ 个数据帧和第 n 个数据帧之间的关系,若存在关系,则设置为 1,若不存在关系,则设置为 0。

当数据帧数据较多时,使用该矩阵表达形式会变得繁琐,因此可进行简写,如拍照指令相关的数据帧,$GL_{32}=(3,4)$,即表示第 3 个通信协议表单中的第 2 个数据帧分别和第 3、4 个数据帧存在关联。数据帧之间的关联一般为指令和数据之间的组合关系。

定义 11. XQ = < SM,GLE >,SM 表示数据帧内部的字段描述; GLE 表示数据帧内的字段之间的关联。

对 XQ 的各子元素定义如下:

定义 12. SM 表示数据帧的详细描述,及字段描述,SM = < Xmn1,Xmn2,…,Xmnk >,其中,k 表示该数据帧内字段的个数,X 代表该字段的字段详细信息。Xmn1 表示通信协议中第 m 张通信协议的第 n 个数据帧的第 1 个字段的内容。

定义 13. X = < < Y,Z > mnk1,< Y,Z > mnk2,…,< Y,Z > $mnkt$; BJ >,其中,t 表示字段 X 的详细信息的个数,Y 为详细信息的文字描述,Z 为详细信息的通信源码,则 < Y,Z > mnk1 表示表示通信协议中第 m 张通信协议的第 n 个数据帧的第 k 个字段的第 1 个信息的文字描述和通信源码,同理,< Y,Z > $mnkt$ 表示通信协议中第 m 张通信协议的第 n 个数据帧的第 k 个字段的第 t 个信息的文字描述和通信源码。若 X 存在边界,则将 BJ 标出,BJ 可用 “( )”或 “【】”来表示开区间或闭区间的上下边界,若无边界,可省略。

定义 14. GLE 表示该数据帧内各字段之间的关联,若该数据帧内字段的个数为 k,$\boldsymbol{GLE}$ 为 $k \times k$ 的上三角矩阵,简称其为二级关联。

$$\boldsymbol{GLE}=\begin{bmatrix} 0 & GLE_{mj12} & GLE_{mj13} & \cdots & GLE_{mj1k} \\ 0 & 0 & GLE_{mj23} & \cdots & GLE_{mj2k} \\ 0 & 0 & 0 & \cdots & \vdots \\ 0 & 0 & 0 & 0 & GLE_{mj(k-1)k} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

其中,GLE_{mj12} 表示第 m 张通信协议表单中的第 j 个数据帧内部的第 1 个字段和第 2 个字段的关系,同理,$GLE_{mj(n-1)k}$ 表示第 m 张表单中的第 j 个数据帧内部的第 $(k-1)$ 个详细数据帧描述字段和第 k 个字段的关系,若存在关系,则设置为 1,若不存在关系,则设置为 0。各字段之间的关联一般为设备状态情况的关联。

建立一级关联和二级关联矩阵显得尤为重要。因为关联矩阵实际为数据帧之间,以及数据帧内字段间的关联,因为有关联,才有字段的有效组合,因为有关联,才剔除了无效冗余的组合测试用例,防止了”组合

爆炸”。

1.2 测试用例生成算法

构建一个如图 1 所示的网络(设 $n=4$,$t1=1$,$t2=3$,$t3=4$,$t4=3$)。

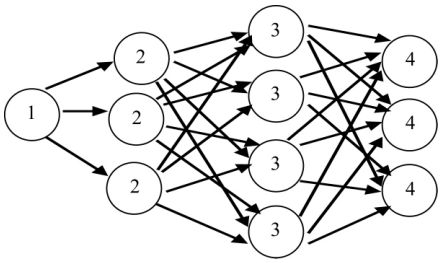

图 1 算法的网络结构图

网络的第 1 层有 1 个节点,表示通信协议表单中的第 1 个数据帧拥有 1 个参数;网络的第 2 层有 3 个节点,表示通信协议表单中的第 2 个数据帧拥有 3 个参数,依此类推。从第 1 层出发,向第 2 层的某一节点推进,直到最后一层的某个节点结束,形成的一条路径,则为一条测试用例。比如 1 -3 -4 -2,表示的是:第 1 个参数取第 1 个详细数据内容,第 2 个参数取第 3 个详细数据内容,第 3 个参数取第 4 个详细数据内容,第 4 个参数取第 2 个详细数据内容。对于用例的生成过程,赋予一个顺序。若按照图 1 的网络结构来看,用例生成顺序应按照从左到右(每一层按照从上到下)的顺序逐个节点执行,若按照路径全覆盖的方式生成,用例数为 $1\times3\times4\times3=36$ 个测试用例。若 1 个表单中有 10 个数据帧,每个数据帧包含 2 个参数,则会有 2^{10}个测试用例。但在工作实际中,1 个通信协议的单个表单中存在的数据帧较多,因此,无法采用路径全覆盖的方法来完成接口测试工作,因为耗费的人力和时间是巨大且不现实的,所以,在实际工作中,需要制定测试用例生成策略。

定义 15. 算法集合 ALG = < SEN,REC >,其中,SEN 表示作为发送端生成测试用例的算法,REC 表示作为接受端生成测试用例的算法。通过判断 JK 的 JKSX 来选择 SEN 和 REC,为保证算法的完整性,以下给出 SEN 算法和 REC 算法。

SEN 算法

输入:基于 IM 输入的发送端通信信息

输出:测试用例集合(CASES)

① 将通信协议表单中作为发送端的某个表单中的每一个数据帧按照帧号顺序,利用 IM 模型框架依次进行输入。

② 设起始帧号 $i=1$,帧长度 $=n$。设数据帧 i 的字段数为 $K(i)$,数据帧 i 的第 x 个字段的详细信息个数为 $T(ix)$,数据帧 i 的二级关联矩阵为 $\boldsymbol{GLE}(i)$,GL_i 为 $\boldsymbol{GLY}$ 的第 i 行,表示数据帧 i 相关联的其他数据帧的关系,GLY_{ij}表示数据帧 i 和数据帧 j 之间的关系;同理,设数据帧 j 的字段数为 $K(j)$,数据帧 j 的第 x 个字段的详细信息个数为 $T(jx)$,数据帧 j 的二级关联矩阵为 $\boldsymbol{GLE}(j)$;设不单独设计测试点的数据帧集合为 NoTest。

③ 判断数据帧 i 内的属性。若数据帧 i 的属性为 COMMAND 或 STATE,则转步骤④,若为 DATA,则转步骤⑭。

④ 判断数据帧 i 的二级关联矩阵 $\boldsymbol{GLE}(i)$ 是否为空;若不为空,则转至步骤⑤;若为空,则转至步骤⑥。

⑤ 将数据帧 i 内的具有关联的字段进行组合,根据 $\boldsymbol{GLE}(i)$ 上三角矩阵的取值,则形成 $\sum_{k=1}^{k(i)}\sum_{h=k+1}^{k(i)}\boldsymbol{GLE}ikh\times T(ik)\times T(ih)$ 个二级组合测试点;转步骤⑦;

⑥ 将数据帧 i 内的 $K(i)$ 个字段形成 $\sum_{k=1}^{k(i)}T(ik)$ 个独立测试点;转至步骤⑧。

⑦ 判断数据帧 i 的一级关联 GL_i 是否为空;若不为空,则转至步骤⑨,若为空,则转至步骤⑩。

⑧ 判断数据帧 i 的一级关联 GL_i 是否为空;若不为空,则转至步骤⑪,若为空,则转至步骤⑫。

⑨ 将数据帧 i 的二级组合测试点和 GL_i 中所有关联的数据帧的每个字段进行两两组合,生成 $[\sum_{k=1}^{k(i)}\sum_{h=k+1}^{k(i)}GLE_{ikh}\times T(ik)\times T(ih)]\times\sum_{\substack{j=1\\j\neq i}}^{n}[GLY_{ij}\times\sum_{k=1}^{k(j)}T(jk)]$ 个一级二级组合测试点,将数据帧 i 没有二级关联的字段和 GL_i 中所有关联的数据帧的每个字段进行两两组合,生成 $[\sum_{k\neq\text{关联的点}}^{k(i)}T(ik)]\times\sum_{\substack{j=1\\j\neq i}}^{n}[GLY_{ij}\times\sum_{k=1}^{k(j)}T(jk)]$ 个一级组合测试点,则总共生成 $[\sum_{k=1}^{k(i)}\sum_{h=k+1}^{k(i)}GLE_{ikh}\times T(ik)\times T(ih)+\sum_{k\neq\text{关联的点}}^{k(i)}T(ik)]\times\sum_{\substack{j=1\\j\neq i}}^{n}[GLY_{ij}\times\sum_{k=1}^{k(j)}T(jk)]$ 个组合测试点,同时,将存在关联的数据帧 j 的帧号放入 NoTest 中,转至步骤⑬。

⑩ 生成 $[\sum_{k=1}^{k(i)}\sum_{h=k+1}^{k(i)}GLE_{ikh}\times T(ik)\times T(ih)]+\sum_{k\neq\text{关联的点}}^{k(i)}T(ik)$ 个测试点,转至步骤⑬。

⑪ 将数据帧 i 中的每一个字段和 GL_i 中所有关联的数据帧的每个字段进行两两组合,生成 $[\sum_{k=1}^{k(i)}T(ik)]\times\sum_{\substack{j=1\\j\neq i}}^{n}[GLY_{ij}\times\sum_{k=1}^{k(j)}T(jk)]$ 个一级组合测试点,同时,将存在关联的数据帧 j 的帧号放入 NoTest 中,转至步骤⑬。

⑫ 生成 $\sum_{k=1}^{k(i)}T(ik)$ 个测试点,转至步骤⑬。

⑬ 将和 i 无关联的数据帧均随机生成有效值,按照数据帧先后顺序将所有数据帧串行连接,依据相应的步骤⑨ ~ 步骤⑫产生的测试点,生成与其测试点数量相同的完整的测试用例,转至步骤⑭。

⑭ i ++,判断 i 是否大于 n,若 i 不大于 n 且 NoTest 不包含 i,则转至步骤③;若 i 不大于 n 且 NoTest 包含 i,则转至步骤⑭重新开始;若 i 大于 n,将所生成的测试用例汇总成该发送端接口的 CASES。

被测配置项作为发送端,需测试其发送的数据帧

是否和通信协议一致，测试发送的数据帧包含状态是否和当前被测配置项一致，只考虑有效类即可。因此，只需按照图 1 的测试用例生成方式（不是全覆盖），将有效类的数据帧按照帧号的顺序连接起来即可。

REC 算法

输入：基于 IM 输入的接收端通信信息

输出：测试用例集合（CASES）

① 将通信协议表单中作为接收端的某个表单中的每一个数据帧按照帧号顺序，利用 IM 模型框架依次进行输入。

② 设起始帧号 $i=1$，帧长度 $=n$。设数据帧 i 的字段数为 $K(i)$，数据帧 i 的第 x 个字段的详细信息个数为 $T(ix)$，数据帧 i 的二级关联矩阵为 $\boldsymbol{GLE}(i)$，GL_i 为 $\boldsymbol{GLY}$ 的第 i 行，表示数据帧 i 相关联的其他数据帧的关系，GLY_{ij} 表示数据帧 i 和数据帧 j 之间的关系；同理，设数据帧 j 的字段数为 $K(j)$，数据帧 j 的第 x 个字段的详细信息个数为 $T(jx)$，数据帧 j 的二级关联矩阵为 $\boldsymbol{GLE}(j)$，数据帧 i 的数据边界测试点个数为 Q_i；设不单独设计测试点的数据帧集合为 NoTest。

③ 判断数据帧 i 内的属性。若数据帧 i 的属性为 COMMAND 或 STATE，则转至步骤④，若为 DATA，则转至步骤⑤。

④ 数据帧 i 的 $T(i)$ 个字段均根据各字段的详细内容生成 1 个异常指令或 1 个异常状态；转至步骤⑥。

⑤ 判断数据帧 i 的数据是否存在边界。若存在上下边界，则生成边界内、上边界上、上边界外、下边界上、下边界外共 5 个数据点，及 $Q_i=5$；若存在上边界或下边界，根据上边界生成上边界内、上边界上、上边界外或是下边界内、下边界上、下边界外共 3 个测试点，及 $Q_i=3$；若不存在边界，则不考虑异常测试点，及 $Q_i=0$；转至步骤⑥。

⑥ $i++$，判断 i 是否大于 n，若不大于 n，则转至步骤③，若大于 n，设 $i=1$，并转至步骤⑦。

⑦ 判断数据帧 i 内的属性。若数据帧 i 的属性为 COMMAND 或 STATE，则转至步骤⑧，若为 DATA，则转至步骤⑨。

⑧ 判断 $\boldsymbol{GLE}(i)$ 是否为空，若不为空，则转至步骤⑩，若为空，则转至步骤⑪。

⑨ 判断 GL_i 是否为空，若不为空，则转至步骤⑫，若为空，则转至步骤⑳。

⑩ 将数据帧 i 内的具有关联的字段进行组合，根据 $\boldsymbol{GLE}(i)$ 上三角矩阵的取值，则形成 $\sum_{k=1}^{k(i)}\sum_{h=k+1}^{k(i)} GLE_{ikh}\times(T(ik)+1)\times(T(ih)+1)$ 个二级组合测试点，随后再加上 $\sum_{k\neq\text{关联的点}}^{k(i)}(T(ik)+1)$ 个独立测试点；转至步骤⑬。

⑪ 将数据帧 i 内的 $K(i)$ 个字段形成 $\sum_{k=1}^{k(i)}(T(ik)+1)$ 个独立测试点；转至步骤⑭。

⑫ 将数据帧 i 和 GL_i 中相关数据帧的每个字段进行两两组合，生成 $Q_i\times\sum_{\substack{j=1\\i\neq j}}^{n} GLY_{ij}\times\begin{cases}\sum_{k=1}^{k(j)}(T(jk)+1), & j\text{ 为指令或状态}\\ Q_j, & j\text{ 为数据}\end{cases}$ 个一级组合测试点；转至步骤⑲。

⑬ 判断数据帧 i 的一级关联 GL_i 是否为空；若不为空，则转至步骤⑮，若为空，则转至步骤⑯。

⑭ 判断数据帧 i 的一级关联 GL_i 是否为空；若不为空，则转至步骤⑰，若为空，则转至步骤⑱。

⑮ 将数据帧 i 的每个字段和 GL_i 中相关数据帧的每个字段进行两两组合，生成 $\left[\sum_{k=1}^{k(i)}\sum_{h=k+1}^{k(i)} GLE_{ikh}\times(T(ik)+1)\times(T(ih)+1)+\sum_{k\neq\text{关联的点}}^{k(i)}(T(ik)+1)\right]\times\sum_{\substack{j=1\\i\neq j}}^{n} GLY_{ij}\times\begin{cases}\sum_{k=1}^{k(j)}(T(jk)+1), & j\text{ 为指令或状态}\\ Q_j, & j\text{ 为数据}\end{cases}$ 个组合测试点，同时，将存在关联的数据帧 j 的帧号放入 NoTest 中；转至步骤⑲。

⑯ 生成 $\left[\sum_{k=1}^{k(i)}\sum_{h=k+1}^{k(i)} GLE_{ikh}\times(T(ik)+1)\times(T(ih)+1)+\sum_{k\neq\text{关联的点}}^{k(i)}(T(ik)+1)\right]$ 个测试点；转至步骤⑲。

⑰ 将数据帧 i 的每个字段和 GL_i 中相关数据帧的每个字段进行两两组合，生成 $\sum_{k=1}^{k(i)}(T(ik)+1)\times\sum_{\substack{j=1\\i\neq j}}^{n} GLY_{ij}\times\begin{cases}\sum_{k=1}^{k(j)}(T(jk)+1), & j\text{ 为指令或状态}\\ Q_j, & j\text{ 为数据}\end{cases}$ 个组合测试点，同时，将存在关联的数据帧 j 的帧号放入 NoTest 中；转至步骤⑲。

⑱ 生成 $\sum_{k=1}^{k(i)}(T(ik)+1)$ 个测试点；转至步骤⑲。

⑲ 将和 i 无关联的数据帧均随机生成有效值，按照数据帧先后顺序将所有数据帧串行连接，依据相应的步骤⑧、步骤⑪～⑭产生的测试点，生成与其测试点数量相同的完整的测试用例；转至步骤⑳。

⑳ $i++$，判断 i 是否大于 n，若 i 不大于 n 且 NoTest 不包含 i，则转至步骤⑦；若 i 不大于 n 且 NoTest 包含 i，则转至步骤⑳重新开始；若大于 n，将所生成的测试用例汇总成该接收端接口的 CASES，再按照各数据帧均为有效值的测试用例生成一个超长数据帧的和一个超短数据帧的超短数据帧。

被测配置项作为接收端，则需按照通信协议的要求响应有效类指令和有效类数据，对于无效类指令采取不响应或对无效类指令进行异常处理，对于异常数据或者超边界的数据采取不响应或进行异常处理。因此，接收端的测试用例采用了有效类分析、无效类分析、边界值等方法来提高接口测试的安全性和可靠性。

2　基于通信协议的接口测试用例生成框架在某软件上的应用

通信协议一般以 Excel 表格形式表示，因为通信协议具有半结构化的特点，因此可直接从表格中提取重点关注的内容。限于篇幅，文中将实际通信协议进行了适当的删减。

2.1　配置项作为输出方

表 1 为某项目中近红外跟踪软件和主控软件的通信接口表单，近红外系统发送给主控数据的数据格式，将通信协议的表单信息输入到 IM，如表 2 所示。

表1 近红外系统发送给主控数据的RS422数据的通信表单

序号	信息名称	BIT								备注
		D7	D6	D5	D4	D3	D2	D1	D0	
1	帧头	0X7E								
2	毫秒	Bit[23:0]								(0.01ms)
3	方位编码器	Bit[31:0]								
4	俯仰编码器	Bit[31:0]								
5	信息属性	默认0000				脱靶量有效:0001 脱靶量无效:0000				
6	目标脱靶量X	Bit[13:0]								(角秒)
7	目标脱靶量Y	Bit[13:0]								
8	自检信息	0000未就位; 1010正常 1111故障				故障类型:相机采集0001; 曝光数据通信0010; 相机设置通信0100; 脱靶量通信1000。				故障类型按bit进行OR操作
9	存储状态	正在存储:0X0A;已停止存储:0XA0;缺省状态:0X55;								
10	相机帧频	1000帧:0X40; 500帧:0X30; 200帧:0X20; 100帧:0X10;50帧:0X01								
11	帧尾	0XE7								

根据IM的输入模型,按照JK,SJZ和XQ进行输入,在XQ中会遇到某些字段内容较多的情况,比如第8个数据帧,可以采用“|”进行字段分隔,为了便于将表2的内容提取出来用于接口测试用例的自动化生成,应按照规定好的分隔符对各部分进行划分。在测试用例自动生成过程中,根据特定的符号,比如“|”、“-”、“;”、“;”、“< >”等符号,通过信息抓取和文本清洗技术[14],可将有用信息直接输入至SEN算法中,同时也可在形成数据帧i的测试点时,将文字描述同时提取出来,可对测试点进行重点描述,有利于测试人员更好地理解和执行测试用例。根据表2的SJZ描述,可以得到一级关联矩阵中$GLY56=1$;$GLY57=1$(因为该事例为1个单独的表单,因此在此将表单号m在实际使用中省略),其余矩阵内的元素均为0。表2中的YY在使用中可以设置为随机值。表3为按照SEN算法自动生成的接口测试用例。对于生成的测试用例,存在着某个测试用例可以替代另一个测试用例的情况,但为了突出每一个测试点,不删除可被替代的测试用例。

表2 按照IM建立的输入框架

序号	按照IM建立的输入框架			
	JK	SJZ	XQ	备注
1		<1,帧头,COMMAND,0>	<帧头-0X7E;0>	
2		<2,微秒,DATA,0>	<微秒-0XYYYYYYY;0>	
3		<3,方位编码器,DATA,0>	<方位编码器-0XYYYYYYYY;0>	
4		<4,俯仰编码器,DATA,0>	<俯仰编码器-0XYYYYYYYY;0>	
5	<近红外	<5,信息属性,STATE,(6,7)>	<(默认-0X0),(脱靶量有效-0X1,脱靶量无效-0X0);0>	
6	发送给主	<6,目标脱靶量X,DATA,0>	<目标脱靶量X-0XYYYY;0>	
7	控的数据帧,SEND	<7,目标脱靶量Y,DATA,0>	<目标脱靶量Y-0XYYYY;0>	YY为任意值
8	>	<8,自检信息,STATE,0>	<(未就位-0X0,正常-0XA,故障-0XF),(相机采集-0X1,曝光数据通信-0X2,相机设置通信-0X4,脱靶量通信-0X8);$\begin{bmatrix}0&1\\0&0\end{bmatrix}$>	
9		<9,存储状态,STATE,0>	<正在存储-0X0A,已停止存储-0XA0,缺省状态-0X55;0>	
10		<10,相机帧频,STATE,0>	<1000帧-0X40,500帧-0X30,200帧-0X20,100帧-0X10,50帧-0X01;0>	
11		<11,帧尾,COMMAND,0>	<帧尾-0XE7;0>	

基于接口参数的黑箱测试用例自动生成算法[12]也是使用接口通信协议内的数据帧作为输入,因此,将其与TCGFICP生成的用例进行比较。同时,将生成的用例与本测评中心从业5年的测试人员设计的用例纳入比较。比较结果见表4。表4中加入了自定义的基于通信协议的黑盒条件组合覆盖率,因为在实际测试中,覆盖率主要在白盒测试上作为指标,比如语句覆盖率、判定覆盖率、条件覆盖率、判定-条件覆盖率、条件组合覆盖率和路径覆盖率这几个指标。而黑盒覆盖率很少,主要基于需求覆盖率。接口测试用例的生成主要基于通信协议,也是需求的一种,结合通信协议的特点和白盒测试的条件组合覆盖率,本文提出一种基于接口的黑盒条件组合覆盖率(黑盒条件组合覆盖率=条件组合至少被评价一次的数量/条件组合总数)来对表4中的3种测试用例生成方法进行比较。

如表4所示,对于TCGFICP生成的测试用例和基于接口参数的黑盒测试用例自动生成算法(简称算法2)以及测试人员设计的用例进行比较。在“针对性”方面,TCGFICP算法在作为发送端在测试用例生成时针对有效类作为用例生成的主线,而算法2没有考虑,因此,无法保证所产生的测试用例最优[12]。在“用例个数”上,TCGFICP算法比测试人员设计的算法多2个(用例1和用例24),因为其他用例包含了帧头和帧尾的验证,但为了突出测试用例的每一个测试点,值得

表 3　自动化生成的近红外系统发送数据给主控的接口测试用例(部分)

序号	接口描述	测试用例
1	近红外发送给主控的数据帧，SEND	帧头(0X7E-0X146CA8-0XFF4587AE-0X009A39A0-0X0-0X1-0X3456-0X10FD-0X0-0X1-0X55-0X40-0XE7)
2		信息属性－脱靶量有效(0X7E-0X686BA7-0XFFFFFFFF-0X00000000-0X0-0X1-0XABCD-0X0975-0X0-0X2-0X55-0X40-0XE7)
3		信息属性－脱靶量无效(0X7E-0XFE0828-0X00000000-0X7F6F4F0A -0X0-0X0-0X0000-0XFFFF-0X0-0X1-0X55-0X40-0XE7)
4		自检信息－未就位－相机采集(0X7E-0X4E5BAC-0X84394793-0X901837572-0X0-0X0-0X4523-0X7FFF-0X0-0X1-0X55-0X40-0XE7)
5		自检信息－未就位－曝光数据通信(0X7E-0X8573DA-0X9864FE14-0X7F6F4F0A-0X0-0X0-0X1010-0XFE70-0X0-0X2-0X55-0X40-0XE7)
6		自检信息－未就位－相机设置通信(0X7E-0X8574DD-0X10030400-0XBCC12987 -0X0-0X0-0XAB43-0X001F-0X0-0X3-0X55-0X40-0XE7)
7		自检信息－未就位－脱靶量通信(0X7E-0X873AAA-0XA0BF8761-0X013450B3-0X0-0X1-0X3481-0X5A5B-0X0-0X4-0X55-0X40-0XE7)
8		自检信息－正常－相机采集(0X7E-0XFEFEFE-0X34565432-0XE2E1F4F7 -0X0-0X0-0X1234-0X0003-0XA-0X1-0X55-0X40-0XE7)
9		自检信息－正常－曝光数据通信(0X7E-0X6BAC21-0X0056DAC0-0X88450000 -0X0-0X0-0X0000-0X0000-0XA-0X2-0X55-0X40-0XE7)
10		自检信息－正常－相机设置通信(0X7E-0X7FFFFF-0X32324532-0XBBCCAADD -0X0-0X0-0XBCBA-0X0214-0XA-0X3-0X55-0X40-0XE7)
11		自检信息－正常－脱靶量通信(0X7E-0X90A24D-0X27FFE350-0X77881AA2 -0X0-0X0-0X3456-0X7FB0-0XA-0X4-0X55-0X40-0XE7)
12		自检信息－故障－相机采集(0X7E-0XCC941A-0XDC5536A2-0X5737ACDA-0X0-0X0-0X9864-0XF675-0XF-0X1-0X55-0X40-0XE7)
13		自检信息－故障－曝光数据通信(0X7E-0X472841-0X05063160-0X9763DCAB-0X0-0X1-0X3532-0XF123-0XF-0X2-0X55-0X40-0XE7)
14		自检信息－故障－相机设置通信(0X7E-0X45DA34-0X987634AC-0X75BBDA23-0X0-0X0-0X14FF-0X2877-0XF-0X3-0X55-0X40-0XE7)
15		自检信息－故障－脱靶量通信(0X7E-0XFAC342-0X010230D0-0X87FFA245-0X0-0X0-0XF2F4-0X34EF-0XF-0X4-0X55-0X40-0XE7)
16－24		……

表 4　作为发送端的 3 种测试用例生成方法的比较

算法	针对性	用例个数	具有关联性的数据帧个数最大值	是否考虑数据帧内的关联性	是否考虑数据类的数据帧的随机性	生成用例的时间	是否会出现重复的用例	黑盒条件组合覆盖率
TCGFICP	有效类	24	3	是	是	小于 1s	可能	100%
基于接口参数的黑盒测试用例自动生成算法	未对数据进行分类	20	2	否	否	小于 1s	否	91%
测试人员设计	有效类	22	3	是	是	30min	否	100%

保留这 2 个测试用例; TCGFICP 算法比算法 2 多 4 个，因为算法 2 没有针对仅包含一个元素的字段的数据帧设计用例。在“具有关联性的数据帧个数最大值”一项中，TCGFICP 可以生成包含 3 个数据帧相互关联的组合测试用例(关联的数量根据实际情况可调整，不受限制)，但算法 2 仅考虑两两组合; 并且，在实际通信协议中，常常出现一个数据帧内包含多个字段，字段包含多个元素的情况，因此，考虑数据帧内的关联性非常重要。对于数据类的数据帧，不适合使用设置好的统一的数据进行测试，因此，在生成测试用例时，数据类的数据帧的随机性也体现出测试数据的多样性。在“生成用例的时间”上，TCGFICP 和算法 2 所用时间相对于测试人员设计使用时间非常小。从实际测试工作的角度出发，为了体现每一个数据帧对应的测试点，TCGFICP 可能会生成重复的用例，但对测试用例生成时间影响很小，可忽略不计。测试人员设计的测试用例通过评审，达到黑盒条件组合覆盖率 100% 的程度，TCGFICP 比测试人员多设计了帧头和帧尾的用例，其他一致，因此 TCGFICP 生成的测试用例可覆盖测试人员设计的用例，因此，达到黑盒条件组合覆盖率 100% 的程度，而算法 2 生成的测试用例的黑盒条件组合覆盖率为 91%。

2.2　配置项作为接收方

表 5 为某项目数字通信发送给粗跟踪小系统的数据格式，通过信息抓取和文本清洗技术，将通信协议的表单信息输入到 IM，如表 6 所示。

表 5　数字通信发送给粗跟踪小系统的数据格式

序号	信息名称	BIT								备注
		D7	D6	D5	D4	D3	D2	D1	D0	
1	帧头	0X7E								
2	调焦控制命令字	按键调焦: 0X0B; 自动调焦: 0XAF; 状态查询: 0XBB;								
3	变倍控制命令字	长焦命令: 0X0D; 短焦命令: 0XD0; 状态查询: 0XDD;								
4	滤光片控制字	0XF1: 1 挡(全透); 0XF2: 2 挡; 0XF3: 3 挡; 0XF4: 4 挡; 状态查询: 0XFF;								
5	调焦命令	正向调焦 +: 7F; 反向调焦 −: FF(按键调焦下有效)								固定调焦步长
6	粗跟踪调焦位置 L	128	64	32	16	8	4	2	1	(12 位 A/D 码)
7	粗跟踪调焦位置 H	×	×	×	×	2048	1024	512	256	
8	帧尾	0XE1								

根据 REC 算法的步骤④，在生成异常测试点时，需要将该字段的描述前加入“异常”进行描述，同样，REC 算法的步骤⑤，存在边界时，在生成边界测试点时，需要将该字段的边界信息加入。表 7 为按照 REC 算法自动生成的接口测试用例。对于可被替代的测试用例，按表 3 方式进行处理。

和算法 2 以及测试人员生成的用例进行比较，比　　较结果见表 8。

表 6　按照 IM 建立的输入框架

序号	按照 IM 建立的输入框架			
	JK	SJZ	XQ	备注
1	<粗跟踪小系统接受数字通讯发送的数据帧，RECI >	<1，帧头，COMMAND，0 >	< 帧头 - 0X7E; 0 >	YY 为任意值
2		<2，调焦控制命令字，COMMAND，(5，6) >	< 按键调焦 - 0X0B，自动调焦 - 0XAF，状态查询 - 0XBB; 0 >	
3		<3，变倍控制命令字，COMMAND，0 >	< 长焦命令 - 0X0D，短焦命令 - 0XD0，状态查询 - 0XDD; 0 >	
4		<4，滤光片控制字，COMMAND，0 >	<1 档 - 0XF1，2 档 - 0XF2，3 档 - 0XF3，4 档 - 0XFF; 0 >	
5		<5，调焦命令，COMMAND，6 >	< 正向调焦 - 0X7F，反向调焦 - 0XFF; 0 >	
6		<6，粗跟踪调焦位置，DATA，0 >	<(粗跟踪调焦位置 H - 0XYYYY; 0X0FFF]); 0 >	
7		<7，帧尾，COMMAND，0 >	< 帧尾 - 0XE1; 0 >	

表 7　自动化生成的粗跟踪小系统接收来自数字通讯数据的接口测试用例(部分)

序号	接口描述	测试用例
1	粗跟踪小系统接收来自数字通讯的数据帧，RECI	帧头(0X7E-0XBB-0X0D-0XFF-0X7F-0X0012-0XE1)
2		异常帧头(0X7A-0XBB-0X0D-0XFF-0X7F-0X0F10-0XE1)
3		调焦控制命令字 - 按键调焦 - 调焦命令 - 正向调焦(0X7E-0X0B-0XD0-0XF1-0X7F-0X0BA2-0XE1)
4		调焦控制命令字 - 按键调焦 - 调焦命令 - 反向调焦(0X7E-0X0B-0XD0-0XF2-0XFF-0X0BA2-0XE1)
5		调焦控制命令字 - 按键调焦 - 异常调焦命令 (0X7E-0X0B-0XD0-0XF2-0X77-0X0AC9-0XE1)
6		调焦控制命令字 - 自动调焦 - 调焦命令 - 正向调焦(0X7E-0XAF-0XD0-0XF1-0X7F-0X0561-0XE1)
7		调焦控制命令字 - 自动调焦 - 调焦命令 - 反向调焦(0X7E-0XAF-0XD0-0XF1-0XFF-0X0000-0XE1)
8		调焦控制命令字 - 自动调焦 - 异常调焦命令 (0X7E-0XAF-0XD0-0XF1-0X77-0X04DE-0XE1)
9		调焦控制命令字 - 状态查询 - 调焦命令 - 正向调焦(0X7E-0XBB-0XDD-0XFF-0X7F-0X0E21-0XE1)
10		调焦控制命令字 - 状态查询 - 调焦命令 - 反向调焦(0X7E-0XBB-0XDD-0XF3-0XFF-0X0399-0XE1)
11		调焦控制命令字 - 状态查询 - 异常调焦命令(0X7E-0XBB-0XDD-0XFF-0X77-0X0AAA-0XE1)
12		异常调焦控制命令字 - 调焦命令 - 正向调焦(0X7E-0XBC-0X0D-0XFF-0X7F-0X0D6C-0XE1)
13		异常调焦控制命令字 - 调焦命令 - 反向调焦(0X7E-0XBC-0X0D-0XFF-0XFF-0X0914-0XE1)
14		异常调焦控制命令字 - 异常调焦命令(0X7E-0XBC-0XD0-0XF1-0X77-0X064B-0XE1)
15		调焦控制命令字 - 按键调焦 - 粗跟踪调焦位置 - 上边界内(0X7E-0X0B-0XD0-0XF1-0X7F-0X0FFE-0XE1)
16		调焦控制命令字 - 按键调焦 - 粗跟踪调焦位置 - 上边界上(0X7E-0X0B-0XDD-0XF2-0XF7-0X0FFF-0XE1)
17		调焦控制命令字 - 按键调焦 - 粗跟踪调焦位置 - 上边界外(0X7E-0X0B-0X0D-0XF3-0XF7-0X1000-0XE1)
18		调焦控制命令字 - 自动调焦 - 粗跟踪调焦位置 - 上边界内(0X7E-0XAF-0XD0-0XF1-0X7F-0X0FFE-0XE1)
19		调焦控制命令字 - 自动调焦 - 粗跟踪调焦位置 - 上边界上(0X7E-0XAF-0XDD-0XF2-0XF7-0X0FFF-0XE1)
20		调焦控制命令字 - 自动调焦 - 粗跟踪调焦位置 - 上边界外(0X7E-0XAF-0X0D-0XF3-0XF7-0X1000-0XE1)
21		调焦控制命令字 - 状态查询 - 粗跟踪调焦位置 - 上边界内(0X7E-0XBB-0XD0-0XF1-0X7F-0X0FFE-0XE1)
22		调焦控制命令字 - 状态查询 - 粗跟踪调焦位置 - 上边界上(0X7E-0XBB-0XDD-0XF2-0XF7-0X0FFF-0XE1)
23		调焦控制命令字 - 状态查询 - 粗跟踪调焦位置 - 上边界外(0X7E-0XBB-0X0D-0XF3-0XF7-0X1000-0XE1)
24		异常调焦控制命令字 - 粗跟踪调焦位置 - 上边界内(0X7E-0XBA-0XD0-0XF1-0X7F-0X0FFE-0XE1)
25		异常调焦控制命令字 - 粗跟踪调焦位置 - 上边界上(0X7E-0XBA-0XDD-0XF2-0XF7-0X0FFF-0XE1)
26		异常调焦控制命令字 - 粗跟踪调焦位置 - 上边界外(0X7E-0XBA-0X0D-0XF3-0XF7-0X1000-0XE1)
27 ~ 36		……
37		异常帧尾(0X7E-0XBB-0X0D-0XFF-0X7F-0X0A17-0XE7)
38		超短帧(0X7E-0XBB-0X0D-0XFF-0X7F-0X0117)
39		超长帧(0X7E-0XBB-0X0D-0XFF-0X7F-0X0001-0XE1-0X10)

表 8　作为接收端的三种测试用例生成方法的比较

算法	针对性	用例个数	具有关联性的数据帧个数最大值	是否考虑数据帧内的关联性	是否考虑数据类的数据帧的随机性	生成用例的时间	是否会出现重复的用例	黑盒条件组合覆盖率
TCGFICP	有效类，无效类，边界值	39	3	是	是	小于 1s	可能	100%
基于接口参数的黑盒测试用例自动生成算法	未对数据进行分类	9	2	否	否	小于 1s	否	24.3%
测试人员设计	有效类，无效类，边界值	37	3	是	是	1h	否	100%

如表 8 所示，仅对“针对性”和“用例个数”进行分析，因为其他比较因素和对表 4 的分析一致。作为接收端的用例设计，需要考虑接收到的异常值、边界值、正常数据帧和异常数据帧的组合、异常和异常数据帧的组合等情况，TCGFICP 的 REC 算法考虑了以上情况，而算法 2 未提出异常处理的思想，因此导致了算法 2 生成的测试用例数远小于 TCGFICP 和测试人员设计的用例数。值得关注的是，TCGFICP 充分考虑了异常情况的组合，例如：第 6、7、9、10 个用例，均为了验证在非按键调焦的情况下触发调焦指令的响应情况；用例 14 为异常调焦控制命令字和异常调焦命令组合的双异常组合；第 15 ~ 17 个用例，均为了测试在按键调焦的情况下在调焦上边界处是否存在异常；第 18 ~ 26 个用例，均为了测试在非按键调焦的情况下结合调焦上边界，是否会出现潜在的异常组合问题。对于接收端的异常处理，和测试人员设计的测试用例保持一致。在“用例个数”上，TCGFICP 比测试人员设计的用例多 2 个，原因和表 3 中的用例 1 和用例 24 一致。测试人员设计的测试用例通过评审，达到黑盒条件组合覆盖率 100% 的程度，TCGFICP 比测试人员多设计了帧头和帧尾的用例，其他一致，因此 TCGFICP 生成的测试用例可覆盖测试人员设计的用例，达到黑盒条件组合覆盖率 100%，而算法 2 生成的测试用例的黑盒条件组合覆盖率为 24.3%。

利用 Visual studio 2010 编写了 TCGFICP 实现的程序，只需将通信协议表单中的信息按照数据帧号依次输入 IM 中，即可自动完成测试用例的设计和编写。文中表 1 和表 5 因为篇幅原因进行了删减，完整的通信协议表单中的数据帧常常高达几十个甚至上百个，会增加测试人员的工作量和测试时间，同时，还可能会出现人为的测试点疏漏。而利用 TCGFICP 生成测试用例，会大大降低测试人员的工作量和测试时间，并且可以消除人为因素造成的测试点遗漏现象。

3　结束语

针对在软件配置项或系统测试中需要耗费大量的人工和时间的问题，在保证接口测试用例质量的前提下，提出了基于通信协议的接口测试用例生成框架 TCGFICP 来自动生成接口测试用例。首先，介绍了基于通信协议的接口测试用例输入模型 IM，通过定义来说明 IM 的三元组结构以及各元素的组成，通过一级关联矩阵建立通信协议表单中数据帧之间的关联，通过二级关联矩阵建立数据帧内字段之间的关联；随后介绍了算法集合 ALG，利用发送端 SEN 和接收端 REC 算法生成测试用例集合 CASES 的过程。

通过对某地面项目的软件发送端和接收端接口的测试用例生成实验，结合信息抓取和文本清洗技术，将通信协议中的信息按照格式要求录入到 IM 中，并分别结合 SEN 算法和 REC 算法，在生成测试用例的同时，提取出了每一个测试用例的测试点内容，并将每一个用例通过关键字描述的形式进行了说明。该方法与人工生成接口测试用例方法相比，能包含人工生成的接口测试用例，虽然存在测试用例出现冗余的情况，但冗余数量相比测试用例总数可以忽略。对于刚刚接触软件测试的人员来说，可以依靠基于通信协议的接口测试用例生成框架 TCGFICP 来完成接口测试用例的设计。该方法符合接口测试用例生成的预期效果，并且提高了工作效率，大大降低了人工成本。

参考文献：

[1] 邢颖. 测试用例自动生成的分支限界算法及实验研究[D]. 北京：北京邮电大学，2011.

[2] Chawla P，Chana I，Rana A. Cloud-based automatic test data generation framework [J]. Journal of Computer and System Sciences，2016，82(5)：712 – 738.

[3] Sofokleous A A，Andreou A S. Automatic，evolutionary test data generation for dynamic software testing [J]. Journal of Systems and Software，2008，81(11)：1883 – 1898.

[4] 姜淑娟，王令赛，薛猛，等. 基于模式组合的粒子群优化测试用例生成方法 [J]. 软件学报，2016，27(4)：785 – 801.

[5] 杨瑞. 基于 EFSM 的测试用例自动化生成关键技术研究[D]. 南京：南京大学，2015.

[6] 丁蕊，董红斌，张岩，等. 基于关键点路径的快速测试用例自动生成方法 [J]. 软件学报，2016，27(4)：814 – 827.

[7] 陈鑫，姜鹏，张一帆，等. 一种面向列车控制系统中安全攸关场景的测试用例自动生成方法 [J]. 软件学报，2015(2)：269 – 278.

[8] Meiliana，Septian I，Alianto R S，et al. Automated test case generation from UML activity diagram and sequence diagram using depth first search algorithm [J]. Procedia Computer Science，2017，116：629 – 637.

[9] Samuel P，Mall R，Kanth P. Automatic test case generation from UML communication diagrams [J]. Information and Software Technology，2007，49(2)：158 – 171.

[10] Wu Y C，Fan C F. Automatic test case generation for structural testing of function block [J]. Information and Software Technology，2014，56(10)：1360 – 1376.

[11] Gutierrez J J，Esalona M J，Mejias M. A model-driven approach for functional test case generation [J]. The Journal of Systems and Software，2015，109：214 – 228.

[12] 聂长海，徐宝文. 基于接口参数的黑箱测试用例自动生成算法 [J]. 计算机学报，2004，27(3)：382 – 388.

[13] 姚香娟，攻敦卫，李彬. 融入神经网络的路径覆盖测试数据进化生成 [J]. 软件学报，2016，27(4)：828 – 838.

[14] 陈虹枢. 基于主题模型的专利文本挖掘方法及应用研究[D]. 北京：北京理工大学，2015.

□