第5卷第5期
1994年5月

软 件 学 报
JOURNAL OF SOFTWARE

Vol. 5, No. 5
May 1994

测试集自动生成工具 TUGEN 的设计与实现*

郝瑞兵　吴建平　　Samuel T. Chanson

（清华大学计算机系，北京 100084）　（不列颠哥伦比亚大学计算机系，加拿大）

摘要　测试集自动生成工具的研究是协议一致性测试领域中比较活跃的一个分支，本文在对目前已有的各种测试集生成方法进行分析的基础上，提出了一种新的测试集自动生成方法并对它的实现 TUGEN 作了介绍. TUGEN 基于一种称为 EBE 的形式模型，EBE 模型只对协议的外部行为进行描述，而且可以从协议的 Estelle 或 LOTOS 描述中转化得到. TUGEN 以协议的 EBE－NF 描述作为输入，使用我们新提出的一套测试事例生成策略，最后产生出 TTCN. MP 格式的测试集. 我们用 X. 25 LAPB 协议的 EBE－NF 描述作为例子，对 TUGEN 的正确性和有效性进行了验证并得到了满意的结果. 我们相信，TUGEN 完全可以成为一个测试集自动生成的有力工具.

关键词　协议一致性测试，被测协议实现(IUT)，测试集，测试事例/测试序列，交互路径，子交互路径.

随着计算机网络和通讯技术的不断进步，通讯协议的设计和实现变得越来越庞大和复杂，而且对于同一协议标准有时会存在多个不同的实现版本. 为了保证协议的各种实现版本之间能够完全地相互访问，最有效的手段就是对这些协议实现进行测试. 协议测试一般可以分为三类：一致性测试、性能测试、互操作性测试. 其中，一致性测试的目的是要验证协议的实现是否与协议标准相一致，它是进行其它两种测试的基础.

目前在协议测试领域中最为活跃且比较成熟的便是协议一致性测试，国际标准化组织 ISO 已经制定了一个有关协议一致性测试的方法和框架的标准(ISO 9646)[1,2]. 基于该标准中所提出的各种用于协议一致性测试的测试结构，我们将被测协议实现 IUT 看作一个黑盒，然后采用局部方法或远程方法将测试事例中所提供的测试输入施加至其上，通过观察由此引起的 IUT 的外部输出来对 IUT 进行测试. 这里所用到的测试事例是由要施加给 IUT 的外部输入和预计中的 IUT 外部输出所组成的输入输出事件序列. 在进行测试时，我们将输入事件加至 IUT 上，然后把观察得到的 IUT 的输出事件与预计的输出事件相比较，如果二者相符，我们便可以认为此 IUT 和协议的形式描述是相一致的；否则，此 IUT 便存在一个或多个错误[8]. 测试集将所有测试事例按照一定的逻辑关系组织在一起. 测试集的生成是

* 本文 1993－08－12 收到

本文是国家自然科学基金(No. 69083318)资助项目. 作者**郝瑞兵**，24 岁，博士生，主要研究领域为计算机网络，协议工程学. **吴建平**，41 岁，教授，主要研究领域为计算机网络，协议工程学，分布式系统. **Samuel T. Chanson**，教授，主要研究领域为计算机网络，协议工程学.

本文通讯联系人：郝瑞兵，北京 100084，清华大学计算机系

© 中国科学院软件研究所 http://www.jos.org.cn

协议一致性测试过程中最为困难和耗时的工作,它一般是由测试专家手工生成的,为了降低测试集生成过程的劳动强度,我们迫切需要用于测试集生成的自动或半自动工具.

由于穷举测试法在实际测试中是不可行的,因此如何从协议的形式描述中自动或半自动地产生出具有较高检错能力和覆盖率同时又不很庞大的测试集,吸引了许多专家学者的注意力.目前已提出的大多数测试例生成方法都是基于某种象有限状态机(FSM)、扩展有限状态机(EFSM)、状态变迁系统等形式模型的,其中一些方法还假定协议是使用 Estelle、LOTOS、SDL 等形式描述语言给出的.通过对这些方法的分析之后,我们发现,它们大都只考虑被测协议实现 IUT 的控制流部分的测试,而对诸如包含协议数据单元(PDU)参数、抽象服务原语(ASP)参数等的数据流部分却基本上没有涉及.近来所提出的几种方法虽然涉及到了数据流部分的测试,但由于其在测试事例生成过程中使用了与协议的内部结构和变量有关的一些信息,从而使得测试事例生成过程变得很复杂.同时,这些方法所生成的测试集在检错能力和执行效率上都不很理想.而且目前现存的这些测试例生成方法所生成的测试事例都是以自己特定的格式表示的,不符合 ISO 9646 所要求的 TTCN 表示方法,因此使得其通用性受到了很大的影响.

在本文中,我们讨论了一个称为 TUGEN 的协议测试集自动生成工具的设计和实现.该工具基于一种称为外部行为描述(EBE)的形式模型,此模型只对协议的外部行为使用输入输出序列及其逻辑关系进行描述.TUGEN 以协议的 EBE 形式描述作为输入,自动产生出 TTCN.MP 格式的测试集.在 TUGEN 中,我们所使用的测试事例生成策略分为以下三步:首先从协议的 EBE 形式描述中区分出所有可能的交互路径 IP;其次通过检查每个 IP 中各个变迁之间的相互依赖关系,产生出子交互路径 SIP;最后由 IP 和 SIP 生成 TTCN.MP 格式的测试集.我们将 TUGEN 应用到 X.25 的数据链路层协议 LAPB 的测试集开发过程之中,并将 TUGEN 产生的测试集与 ISO/IEC 所提供的测试集(DIS 8882 的第二部分)做了比较,发现我们的测试事例生成方法是简单和正确的,它所产生出的测试集具有较高的检错能力和执行效率.我们相信,TUGEN 可以作为一个强有力的支持工具应用到协议一致性测试集的开发过程中去.

本文第 1 节主要介绍 TUGEN 工具的理论基础 EBE 模型及基于 EBE 模型所构造的测试集生成算法.第 2 节主要讲述 TUGEN 的设计与实现.对 TUGEN 的各个主要模块的功能及数据在各个模块之间的流动过程做了详细介绍.在第 3 节我们通过使用 X.25 LAPB 的测试集开发为例,对 TUGEN 的正确性和有效性进行了验证与评价.本文最后一节对 TUGEN的设计和开发工作进行了总结.

1　TUGEN 的理论基础

TUGEN 的理论基础是一种称为外部行为描述(External Behaviour Expression)的形式模型[3].在利用现有的形式描述技术如 Estelle 和 LOTOS 来描述协议时,一般都会包含一些协议的内部结构或者变量[5—7].我们认为这些与协议的内部结构相关的信息的使用会使测试事例的自动生成过程复杂化.另一方面,这些协议的内部结构对于协议状态变迁的影响完全可以通过外部可观察到的行为来表示.因此,我们提出了一种只对协议的外部行为进

© 中国科学院软件研究所 http://www.jos.org.cn

行描述的形式模型 EBE，作为我们的测试序列自动生成工具的理论基础.

1.1 EBE 模型的基本定义

EBE 模型化一系统的外部可见行为时，要使用该系统与外部环境之间所交换的交互作用序列和这些序列中的元素(输入/输出原语及其参数)间的逻辑关系. 下面我们给出 EBE 模型的定义.

定义 1. 一个外部行为描述(EBE)是一个四元式，EBE=(S,s_0,T,R)，其中：

S 被模型化系统的外部状态的有限集

$s_0 \in S$ 被模型化系统的初始外部状态

T 外部状态之间的变迁集

R 变迁之间的逻辑关系集

系统的外部状态是系统和外部环境交换交互作用时的一个暂停状态.

定义 2. 一个外部状态之间的变迁是系统和外部环境之间的一交互作用，它由输入/输出原语及其参数所组成. 变迁的一般形式为 $T_{ij}=\langle I_p,O_q\rangle$，其中 $T_{ij}\in T,I_p\in I,O_q\in O$：

(1) I 是外部环境的输入原语集合，其中每个输入原语格式为：$I_p(Xp_1,\cdots,X_{pn})$，I_p 为输入原语标识符，$X_{p1},\cdots,X_{pn}(n\geq 0)$是输入原语 I_p 的参数.

(2) O 是系统对外部环境的输出原语集合，其中每个输出原语格式为：$O_q(Yq_1,\cdots,Y_{qm})$，O_q 为输出原语标识符，$Y_{q1},\cdots,Y_{qm}(m\geq 0)$是输出原语 O_q 的参数.

(3) 当无输入原语或输出原语时，用“—”表示.

定义 3. 逻辑关系集 $R_{ij}=(F,P)$成立的充要条件是存在一从状态 s_i 到状态 s_j 的变迁 T_{ij}.

(1) Z 是一组对变迁 T_{ij}的输出有影响的元素集，其中的元素一般为输入原语及其参数，也可能是系统实现申明中的协议参数.

(2) F 是对应变迁 T_{ij}的函数集合，当且仅当存在满足函数集 F(Z)的 Z 时，即$\{Y_{q1},\cdots,Y_{qm}\}=F(Z)$，才会产生 T_{ij}的输出参数 $Yq_1,\cdots,Y_{qm}(m\geq 0)$.

(3) P 是对应变迁 T_{ij}的谓词条件集合，当且仅当存在满足 P(Z)的 Z 时，变迁 T_{ij}才会发生.

定义 2 和定义 3 用来模型化一个系统的外部行为，并对输入/输出原语的参数的值及其变化进行处理. 除了逻辑关系 F 和 P 外，在变迁 T_{ij}中有时会发生一些操作，这些操作主要用于处理系统全局变量及完成一些系统所必需的动作.

1.2 EBE 模型的描述方法

EBE 模型的描述方法共有两种：一种方法是使用一具有树结构的有向图来描述，我们称之为基于行为树的 EBE (EBE－BT)；另一种方法称为基于规范式的 EBE (EBE－NF). 在 EBE－BT 中，树节点表示一个系统的外部可观察状态(即状态集合 S)，树根为系统的初始状态 s_0，连接树节点的树枝表示 EBE 的外部状态之间的变迁(即集合 T). 和一个变迁相关联的逻辑关系可以用附加说明的方式来描述. 这样，在 EBE－BT 和 EBE 之间便有一个明确的映射关系. 在 TUGEN 中，我们使用 EBE－BT 作为内部数据结构来存储包含在协议的 EBE 描述中的信息.

EBE－NF 是使用 EBE 来描述一个系统的另一方法. 一个系统的 EBE－NF 描述的主

© 中国科学院软件研究所 http://www.jos.org.cn

框架如下:

```
SPECIFICATIONprotocol_name
    [输入/输出原语及其参数表]
CONST
    系统常量定义表
VAR
    系统变量定义表
FUNC
    系统全局函数定义表
BEHAVIOUR
    系统的外部行为描述部分
ENDSPEC
```

系统的外部行为描述是 EBE－NF 中最重要的一部分,我们可以使用 EBE－NF 的语法规则来构造这一部分. EBE－NF 的语法规则可以使用 BNF(巴克斯范式)进行描述. 由于 EBE－NF 只对系统的外部行为进行描述,因而它的语法较 Estelle 和 LOTOS 来讲都简单.

一个协议的 EBE－NF 描述可以从协议文本直接得到,然而我们更感兴趣的是由协议的 Estelle 或 LOTOS 描述中直接生成 EBE－NF 描述[4]. EBE 的结构与 Estelle 非常类似,EBE 的树结构(包括状态和变迁)可以直接从 Estelle 的有限状态机中得到,而逻辑关系部分则可通过搜索 Estelle 描述中每个变迁的操作部分而获得. 因此,利用一些转化规则,我们可以从协议的 Estelle 描述中直接得到 EBE 描述. LOTOS 的操作语义使得我们能够从协议的 LOTOS 描述中构造出协议的变迁树,该树和 EBE－BT 的结构是相同的,EBE 的逻辑关系部分可以通过检查 LOTOS 描述中的数据结构和值表达式来获得. 因此,我们也可以从协议的 LOTOS 中获得 EBE 描述. 现在,我们正在开发一个由 LOTOS 到 EBE 的转化工具.

1.3　TUGEN 中的测试事例生成策略

在[1,2]中,协议一致性测试所需的测试序列的集合称为测试集. 测试集具有层次化的结构,其基本单元为测试事例. 每个测试事例都有非常明确的测试目的. ISO 已经提出了一种测试集的形式化描述方法,即树表组合表示法(TTCN). 我们的 TUGEN 所产生出的测试集便是以 TTCN 格式描述的. 在 TUGEN 中,我们采用[3]中所提出的测试事例生成策略并对其做了一些修改和扩充. TUGEN 中所使用的测试事例生成策略可分为三步,我们将在下面分别予以详述,其核心思想是对每一输出原语(或输出原语参数)和对它有影响的输入原语(或输入原语参数)之间的关系进行测试. 假定 I_p 为一参数为 $X_{p1},\cdots,X_{pn}(n\geqslant 0)$ 的输入原语,O_q 为一参数为 $Y_{q1},\cdots,Y_{qm}(m\geqslant 0)$ 的输出原语,当在协议的某一状态外部输入为 I_p 时会产生输出 O_q,我们称 I_p 对 O_q 有影响;当 X_{pi} 是使输出原语 O_q 的谓词条件 P(Z) 成立的 Z 中的一个元素时,我们称 X_{pi} 对 O_q 有影响;当输出原语参数 Y_{qj} 是通过对输入原语参数 X_{pi} 的某种操作得到的时,我们称 X_{pi} 对输出原语参数 Y_{qj} 有影响.

1.3.1　交互路径的生成(Step1)

定义 4.(InteractionPath)

交互路径(IP)是协议实现与外部环境之间相互交换的一系列交互作用的记录轨迹,它始于协议的初始外部状态 s_0,并终止于同一状态. 任何循环的交互作用在交互路径(IP)中只游历一次. 当 IP 中存在一个或多个循环的交互作用时,我们将其称为带循环的交互路径

© 中国科学院软件研究所 http://www.jos.org.cn

(IP).

基于协议的 EBE—NF 描述,我们使用[3]中的算法 A 来找出所有的交互路径.由于篇幅所限,本文不列出算法的详细内容.

1.3.2 输入/输出子路径的生成(STEP2)

在得到所有的交互路径之后,我们使用[3]中描述的算法 B 来得到所有的输入/输出子路径(SIP).每一 IP 包含一个或多个 SIP,每个 SIP 可以从属于一个或多个 IP.每一 SIP 带有一个或多个测试目的.在算法 B 中,我们将初态和终态相同的变迁称为自循环变迁.若在某个 IP 中包含一个或多个自循环变迁,则将此 IP 称为自循环 IP.关于算法 B 的详细描述可参见[3],本文只给出一些要用到的基本概念的定义.

定义 5. (**I/O Subpath**)

一条输入/输出子路径(SIP)是 IP 中的一段交互作用的记录轨迹 $e_1,\cdots,e_k$,其中:

(1) e_i 是一参数为 $X_{i1},\cdots,X_{in}(n\geqslant 0)$的输入原语 I_p,e_k 是一参数为 $Y_{k1},\cdots,Y_{km}(m\geqslant 0)$的输出原语 O_k.

(2) e_k 的逻辑关系(包括谓词条件 P(Z)和函数关系 F(Z))得到满足.

(3) e_1 是 IP 中包含 Z 中元素的最先的一个变迁.

定义 6. (**SIP 的测试目的**)

每一 SIP 的测试目的是它所对应的在定义 5 中定义的交互序列 $e_1,\cdots,e_k$ 和 e_k 的参数 $Y_{ki}(m\geqslant i\geqslant 0)$.

1.3.3 TTCN 格式测试集的生成(STEP3)

测试事例是一个为了获得某一特定的测试目的所需的输入输出动作的完整的独立的描述,它的起始和终止状态都应为稳定的状态.在 STEP2 中所产生的 SIP 只是在测试执行过程中的正确的交互作用执行序列.基于 SIP,我们可以构造出含有正确性、不确定性和防护性分支的测试树,并对树中的每一个叶节点赋予三种测试判决之一.每个测试树可以完整的映像到一个 TTCN 格式的测试事例.将测试事例安照一定的规则予以分组,便可以得到完整的 TTCN 格式的测试集.

一.从 SIP 生成 TTCN 格式的测试事例

定义 7. (**TestTree**)

一个测试树是对应于一特定测试目的的由输入或输出事件组成的多分支树,其每一分支标以带参数的输入或输出原语.若树中每个输入事件只对应单一的输出事件,则称此树为确定性的;若树中存在一个输入事件对应于多个输出事件的情况,则称此树为不确定性的.

定义 8. (**amble**)

一个状态的"前件"(preamble)定义为从初始状态 s_0 到此状态的一条最短的可执行路径.一个状态的"后件"(postamble)定义为从此状态到初始状态 s_0 的一条最短的可执行路径.

我们首先将每一 SIP 转化为一棵测试树,并对能够合并的测试树进行合并.测试树中可包含三种分支:即正确性分支,它是此测试树的主要测试目的;非确定性分支,它属于正确的协议行为,但不是此测试树的主要测试目的;防护性分支,它一般为协议本不该发生的行为.最后生成的测试树只有两种:确定性的和非确定性的.确定性的测试树中不含有非确定

© 中国科学院软件研究所 http://www.jos.org.cn

性分支.

对于测试树中的每一个叶子可以赋予三种测试判决之一.正确性分支的叶子其测试判决为 PASS;非确定性分支的叶子其测试判决为 INCONCLUSIVE;防护性分支的叶子其测试判决为 FAIL.

对于 EBE－NF 中的每一个状态找出其相应的前件和后件并赋予不同的标识符以备引用.

按照 TTCN 的语法,将每个测试树转化为一 TTCN.MP 格式的测试事例.

二.生成 TTCN 格式的测试集

测试事例组是一组相互关联的测试事例的集合.在 TUGEN 中,我们将所有初态相同的测试事例归在同一组中.

我们测试集产生方法的最后一步是生成一 TTCN 格式的测试集.该测试集的测试集概述部分、说明部分及约束部分都可由包含在协议的 EBE－NF 描述中的信息直接生成.它的动态部分则正是我们在上面所生成的测试组.这样,一个完整的 TTCN 格式的测试集便从协议的 EBE－NF 描述中产生了.

2　TUGEN 的设计与实现

2.1　TUGEN 的总体结构

基于前一章中所提出的测试事例生成算法,我们使用 C 语言在 SUN 工作站上实现了一个测试集自动生成工具 TUGEN.TUGEN 的总体结构参见图 1,它由五个模块组成,这些模块分别为:EBE 语法分析器,IP 生成模块,SIP 生成模块,TTCN 测试集生成模块,公共支持模块.这五个模块由一个称为 TUGEN 的主控模块所控制.

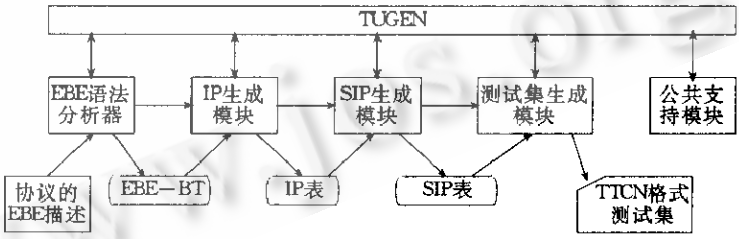


图1　TUGEN总体结构图

各个模块之间的数据流动关系为:

(1) 给定一个协议的 EBE－NF 描述,EBE 语法分析器模块将其转换为 EBE－BT 形式的内部数据结构.

(2) IP 生成模块从 EBE－BT 中找出所有可能的交互路径 IP 并将其存入 IP 表中.

(3) SIP 生成模块从上一步所产生出的 IP 中生成出所有的 SIP 并存放在 SIP 表中.

(4) 基于协议的 EBE－NF 描述和 SIP 表,产生 TTCN 格式的测试集作为最后的输出.

2.2　TUGEN 中的基本数据结构

© 中国科学院软件研究所 http://www.jos.org.cn

TUGEN 中的主要数据结构为两个表,即存储从协议的 EBE－NF 中得到的信息的变迁(Transition)表和状态(State)表,TUGEN 中各个模块的功能都是在对这两张表进行操作的基础上完成的.

一. 变迁(Transition)表

EBE 语法分析器将从协议的 EBE－NF 描述中抽取出的信息存放在变迁表中,变迁表中的每一项代表协议的 EBE－NF 描述中的一个变迁,该记录项包含了协议的 EBE－NF 描述中每一变迁的详细信息,如变迁的起始状态、终止状态、输入输出原语及其参数等等.变迁表中的每项大小是一样的,这使得我们能够对变迁表进行随机的存取访问.变迁信息中大小不固定的部分我们只在变迁项中存储一指针,指向存放实际信息的动态分配的存储块.在变迁表的定义中我们使用了一种称为通用表(General Table)的表结构来存放各种元素组,这种通用表的定义如下:

```
typedef struct{
    char   *key;   /*元素名称*/
    int    type;   /*元素类型*/
    int    ref;    /*引用此元素的变迁索引*/
    char   *pur;   /*引用此元素的目的*/
}TABREC;           /*通用表记录项定义*/
typedef struct{
    int   size;    /*通用表的最大元素数*/
    int   num;     /*通用表中实际元素数*/
    int   sorted;  /*表排序与否标志*/
    TABREC  **table;  /*元素记录表*/
}TABLE;            /*通用表的定义*/
```

TUGEN 中提供了对通用表进行各种操作的子过程,它们是构成变迁表操作的基础.

二. 状态(State)表

利用变迁表中各个变迁的出发态和结束态之间的相互联系,我们可以构造出与 EBE－BT 形式相同的变迁树,在 TUGEN 中,变迁树使用一个单独的状态表来存放.状态表中的每一项对应于协议的一个状态,其大小是固定的,由于从各个状态出发的变迁数是不等的,所以这一部分用动态分配的数组来存放.变迁表和状态表之间的关系可以用图 2 中的例子来说明.

2.3 TUGEN 中各个模块的功能及实现

在这一节中,我们对 TUGEN 中各个模块的主要功能及其实现方法作一简要介绍.

一. EBE 语法分析器

此模块的输入为协议的 EBE－NF 描述,输出为协议的 EBE－BT(存放在变迁表和状态表中).它包含一个词法分析器和一个语法分析器.词法分析器将输入拆成一个一个的"token",为了简化起见,词法分析器返回的是字符串形式的"token",而不象一般的分析器那样返回的是"token"的编号值.语法分析器使用这些"token"和 EBE－NF 的产生式语法来生成协议的 EBE－BT 并存放在变迁表和状态表中.

二. IP 生成模块

这个模块用于从 EBE－BT 中找出所有可能的交互路径 IP,它是我们的测试生成策略

© 中国科学院软件研究所 http://www.jos.org.cn

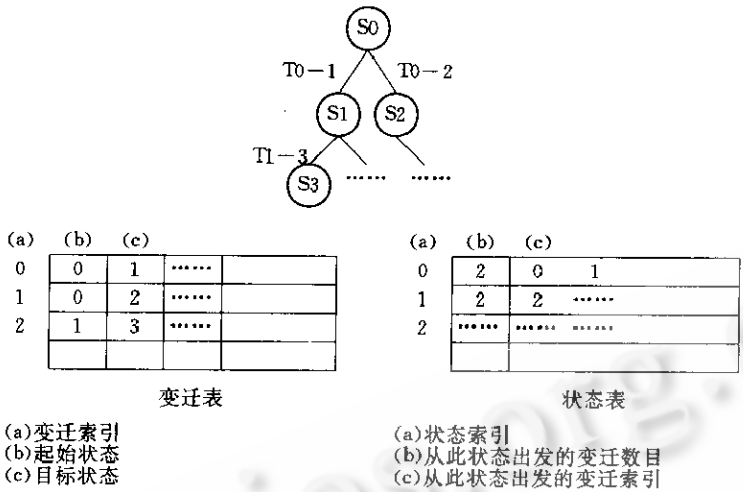


图2 变迁表与状态表的关系

中算法 A 的具体实现. 在实现过程中,我们使用了递归过程调用以生成所有可能的交互路径,其实现的大框架如下:

(1)使用参数(IP=NULL,NODE=根节点)调用第(2)步.

(2)给定一条 IP 和一个树节点 NODE:

若 NODE 是叶子节点,则此 IP 已结束,将其存放于 IP 表中. 否则,对于从该节点 NODE 的出发的每一个变迁,作以下判断:若此变迁不是循环变迁,对原 IP 作一拷贝并将此变迁附加上得到一新的 IP,使用参数(IP=新 IP,NODE=此变迁的目的节点)递归调用第(2)步.

若此变迁为循环变迁,将所有同组的循环变迁附加在当前 IP 的末尾,并寻找一回到初始状态 s_0 的路径附加至此 IP,将此 IP 存放于 IP 表中,并设置循环 IP 的标志.

三. SIP 生成模块

此模块是我们在第 1 节中所提出的算法 B 的实现,它基于 IP 生成模块所产生的 IP 表,通过对每一条交互路径中各个变迁之间相互依赖的检查,生成所有的子交互路径并存放于 SIP 表中.

如何确定交互路径中的变迁之间的相互依赖关系是一件比较困难的事情. 当某个变迁的输出原语和参数受到另一个变迁的输入原语、输入原语参数、协议实现参数(在协议实现申明中定义并在此变迁中被引用)的影响时,我们称第一个变迁依赖于第二个变迁. 为了找出变迁之间的这种依赖关系,我们为每一个变迁构造了两个通用表,RHSref_table 表和 inparm_table 表. RHSref_table 表中存放在定义 3 中所定义的集合 Z 中的所有元素,即对该变迁有影响的元素. inparm_table 表中存放此变迁的输入原语参数和一些在此变迁中引用到的协议实现参数. 在确定两个变迁之间是否存在依赖关系时,我们检查第二个变迁的 RHSref_table 表和第一个变迁的 inparm_table 表,看是否在两个表中的元素之间存在一些对应关系. 若有的话,则称第二个变迁依赖于第一个变迁,它们属于同一个子交互路径 SIP.

四. TTCN 格式测试集生成模块

© 中国科学院软件研究所 http://www.jos.org.cn

此模块基于协议的 EBE 描述和 SIP 生成模块所产生的子交互路径(SIP),生成一个使用 TTCN 描述的测试集.它使用我们在 1.3.3 中所提出的策略.

在实际执行中,我们发现会有多个路径同时满足成为 preamble 或 postamble 的条件,我们采取了选取第一个满足条件的路径成为 preamble 或 postamble 的措施.

在将 SIP 转化为测试树的过程中,我们发现在同一测试事例组中,有些测试树具有相同的前缀,即从测试树的根部开始,有一段输入/输出原语及其参数是相同的,因此我们有必要对这些测试树做一合并,只产生一个多分支的测试树作为最后的测试事例.

五.公共支持过程

本模块提供了几个上面四个模块都需要的公共子过程.它们主要完成对内部数据结构如各种表的存取.我们将这些子过程从各个模块中提取出来的原因是想提供一良好的数据封装性,这是面向对象的程序设计中的基本技术之一.

六.主控模块 TUGEN

主控模块 TUGEN 协调各个子模块之间的关系使它们能够顺利的完成测试集生成任务.另外,本模块还提供了一些使用 TUGEN 的帮助信息.

3 TUGEN 的验证和评价

在本节中,我们使用 X.25 的 LAPB 协议的 EBE 描述作为例子,对 TUGEN 的正确性及效率进行验证和评价.

3.1 X.25 LAPB 协议的 EBE 描述

为了保证数据设备的兼容性,CCITT 提出了一个称作 X.25 的建议作为分组交换网中数据电路终端设备(DCE)和分组方式的数据终端设备(DTE)之间的标准接口协议.LAPB(平衡式链路访问规程)是 X.25 建议中的数据链路层协议.由于在 X.25 LAPB 协议标准文本中并没有明确的给出状态的定义,所以我们首先给出该协议的测试状态定义:

S0—拆接状态

S1—链路正在拆接且计时器 T1 已启动的状态

S2—链路正在建立且计时器 T1 已启动的状态

S3—正常数据传送状态

S4—FRMR 监控帧已发出的状态

S5—RNR 监控帧已发出的状态

S6—REJ 监控帧已发出的状态

基于上面给出的七个状态及 X.25 LAPB 协议的文本,我们可以得到 X.25 LAPB 的 EBE 描述,这里只摘录出其中描述状态 S0 的行为的那段 EBE—NF.

```
……
S0   [DISC(p=0),DM(f)|
       FUNCTION:DM.f=0;
       PREDICATE:true;]*S0{T0—0(1)|R0—0(1)}
   +[DISC(p=1),DM(f)|
```

© 中国科学院软件研究所 http://www.jos.org.cn

```
    FUNCTION:DM.f=1;
    PREDICATE:true;]*S0{T0-0(2)|R0-0(2)}
  +[SABM(p=0),DM(f)|
    FUNCTION:DM.f=0;
    PREDICATE:true;]*S0{T0-0(3)|R0-0(3)}
  +[SABM(p=1),DM(f)|
    FUNCTION:DM.f=1;
    PREDICATE:true;]*S0{T0-0(4)|R0-0(4)}
  +[DM(f=0),-|
    FUNCTION:nil;
    PREDICATE:true;]*S0{T0-0(5)|R0-0(5)}
  +[DM(f=0),DISC(p)|
    FUNCTION:DISC.p=1;
    PREDICATE:true;]*S1{T0-1(1)|R0-1(1)}
  +[DM(f=0),SABM(p)|
    FUNCTION:SABM.p=1;
    PREDICATE:true;]*S2{T0-2(1)|R0-2(1)}
  +[SABM(p=0),UA(f)|
    FUNCTION:UA.f=0;
    PREDICATE:true;]*S3{T0-3(1)|R0-3(1)}
  +[SABM(p=1),UA(f)|
    FUNCTION:UA.f=1;
    PREDICATE:true;]*S3{T0-3(2)|R0-3(2)}
.....
```

为了简化起见,我们只对协议的基本部分进行了描述,即在每一状态只对接收到正确帧和不适宜帧时的系统外部行为进行描述,而省略了对错误帧的处理.

3.2 X.25 LAPB 协议测试集生成过程

我们将 TUGEN 用于 X.25 LAPB 协议测试集的开发过程,总共产生交互路径 IP 1229 条,以下为其中的若干条 IP:

IP[1]:S0,T0-0(1),S0

IP[2]:S0,T0-0(2),S0

IP[5]:S0,T0-1(1),S1,T1-0(1),S0

IP[6]:S0,T0-1(1),S1,T1-0(2),S0

IP[10]:S0,T0-1(1),S1,T1-1(1),S1,T1-1(2),S1,T1-0(1)

从这些 IP 中,我们总共得到子交互路径 SIP 85 条,以下为其中的若干条 SIP(每条 SIP 的测试目的在此省略):

SIP[1]:T0-0(1)

SIP[2]:T0-0(2)

© 中国科学院软件研究所 http://www.jos.org.cn

SIP[3]:T0—0(3)

SIP[4]:T0—0(4)

SIP[5]:T0—0(5)

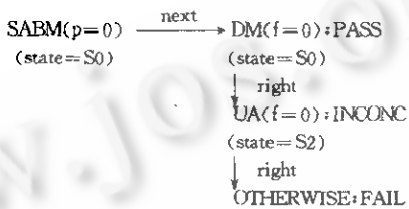
SIP[6]:T0—1(1)

SIP[7]:T0—2(1)

SIP[8]:T0—3(1)

SIP[9]:T0—3(2)

我们将这些 SIP 转化为测试树,并对"前缀"相同的测试树进行合并,下面是一合并后的测试树:



我们对这些测试树按照其出发状态进行分组,共得到七个测试事例组,然后为每一测试事例组生成一个测试事例前件(preamble)和一个测试事例后件(postamble). 下面是测试事例组 2(对应于状态 S1)的前件和后件:

前件 preamble[2] :T0—1(1)

后件 postamble[2] :T1—0(1)

最后,我们按照 TTCN 测试集的格式生成 X. 25LAPB 协议的 TTCN. MP 格式的测试集. 在结果测试集中,上面的测试树对应以下的测试事例:

```
Testcase[3]:
    +preamble—0
      !SABM(p=0)
       ?DM (f=0)   PASS
            +postamble—0
       ?UA(f=0)   INCONC
       ?OTHERWISE FAIL
```

测试集的测试集概述部分、说明部分都是依据协议的 EBE—NF 描述部分所包含的信息生成的.

3.3 对 TUGEN 的验证与评价

为了验证 TUGEN 的正确性和评价其有效性,我们将由 TUGEN 所产生的 X. 25 LAPB 的测试集和 ISO/IEC X. 25 LAPB 的标准测试集(DIS 8882的第二部分)作了一个比较. 为了叙述的方便,在下面的讨论中,我们用"测试集 LAPB"来表示由 TUGEN 所产生的测试集,用"标准测试集"表示 DIS 8882中的测试集.

由于 TUGEN 所采用的测试事例分组方法与标准测试集相同,即所有出发状态相同的测试事例属于同一测试事例组,所以这两个测试集具有一定的可比性. 在表1中我们给出了二者的对比.

© 中国科学院软件研究所 http://www.jos.org.cn

表1　两种测试集动态部分的比较

组号	测试集 LAPB	标准测试集(基本部分)
1	5	5
2	7	7
3	7	7
4	25	21
5	8	6
6	14	12
7	12	11

通过比较，我们得到以下结论：

(1)测试集 LAPB 的测试集概述部分、说明部分和标准测试集是基本一致的，但是它没有约束部分，这部分主要是各种 PDU 的编码，由于用机器处理难度较大，所以我们将其留下用人工生成.

(2)测试集 LAPB 的动态部分覆盖了标准测试集的动态部分的基本测试事例(即对系统收到正确帧和不适宜帧时的行为进行测试的测试事例). 我们的 TUGEN 还产生了一些标准测试集中不包括的测试事例，这是因为我们的测试事例生成策略特别强调变迁之间的相互依赖关系，所以在测试集 LAPB 中将一些在标准测试集中作为单个测试事例的变迁的组合作为测试事例来对待. 我们认为这些测试事例是正确的，在进行测试时应当予以执行.

(3)某些标准测试集中的测试事例未在测试集 LAPB 中出现，其原因是这些测试事例是用于测试 IUT 的防护性的，即当 IUT 收到不正确的帧时所做出的反应. 由于在 EBE—NF 中对这些行为进行描述会导致大量循环变迁的产生，这将直接影响测试集的生成效率，所以我们在目前的实现版本中忽略了对这些行为的描述和处理.

总的来讲，TUGEN 基本上实现了从协议的形式化描述中自动生成测试集的目标. 它所产生出的测试事例是正确的和完善的，同时也是有效的.

结　论：正如上面几节所描述的，TUGEN 是一用于从协议的形式描述中自动生成测试集的工具，它所基于的协议的 EBE 描述可由协议的 Estelle 或 LOTOS 描述直接转化得到. TUGEN 的理论基础是吴建平老师在加拿大 UBC 大学做访问学者期间提出的[3]，在实际实现中我们对它作了一些修改和扩充. 我们将 TUGEN 应用到 X.25 LAPB 协议和 TP0协议的测试集开发工作中，取得了令人满意的结果，同时也对 TUGEN 的正确性和有效性做了验证和评价. 我们相信 TUGEN 可以为协议的一致性测试过程提供很大的便利.

致谢　加拿大 UBC 大学的 Dennis 和 Stuart 曾经开发了此工具的早期版本，他们卓有成效的工作成为 TUGEN 开发工作顺利完成的基础. 在此仅表示真诚的感谢！

参考文献

1　ISO/IEC. OSI conformance testing methodology and framework, part 1: general concepts. IS 9646—1 First Edition, 1991.

2　ISO/IEC. OSI conformance testing methodology and framework, part 3: the tree and tabular combined notation (TTCN). DIS 9646—3, 1990.

3　Wu J P, Chanson S T. Test sequence derivation based on external behaviour expression. Proc. of 2nd International Workshop on Protocol Test Systems, Berlin(West), Germany, Oct 1989.

© 中国科学院软件研究所 http://www.jos.org.cn

4 Wu J P, Chanson S T. Translation from LOTOS and specification to extended transition system and its verification. Proc. of 2nd International Conference on Formal Description Techniques for Distributed System and Communications Protocols, Vancouver, Canada, 1989.

5 Behcet SARIKAYA. Conformance testing: architectures and test sequences. Computer Network and ISDN Systems, 1989;(17).

6 Lee D Y, Lee J Y. A well-defined Estelle specification for the automatic test generation. IEEE Transaction on Computers, 1991;40(4).

7 Dam H V, loosterman H K, Kwast E. Test derivation for standardized test methods. Protocol Test Systems, IV.

8 Rayner D. Standardizing conformance testing for OSI. Proc. of 5th IFIP/WG6.1 International Workshop on Protocol Specification, Testing and Verification, Toulouse-Moisssac, France, June 1985.

9 Sheriff M H, Hoover G L. X.25 conformance testing—a tutotrial. IEEE Communications Magazine, 1986;24(1).

TUGEN: A TOOL FOR AUTOMATIC TEST SUITE GENERATION

Hao Ruibing and Wu Jianping

(*Department of Computer Science, Tsinghua University, Beijing* 100084)

Samuel T. Chanson

(*Department of Computer Science, University of British Columbia Vancouver, Canada*)

Abstract This paper presents a tool called TUGEN which is used for automatic test suite derivation from formal protocol specification. TUGEN is based on a formal model called EBE (External Behavior Expression) which can be obtained from formal protocol specification in either Estelle of LOTOS. This model specifies only the external behavior of a protocol in terms of the input/output sequences and their logical (function and predicate) relations. Based on the EBE specification of a protocol, a test sequence derivation method is used to identify associations between inputs and outputs through the interaction paths and their I/O subpaths, then generic test cases specified in TTCN (Tree and Tabular Combined Notation) can be generated from these I/O subpaths. Comparison of test cases generated from this tool and those in ISO/IEC DIS 8882 part 2 for X.25 LAPB protocol shows that the resulting set of test cases of TUGEN is concise and effective. It is our belief that TUGEN can be a powerful utility for protocol test suite generation.

Key words Protocol conformance testing, implementation under testing, test suite, test case/test sequence, interaction path, sub-interaction path.

© 中国科学院软件研究所 http://www.jos.org.cn