文章编号：0253-374X(2015)03-0416-07

DOI:10.11908/j.issn.0253-374x.2015.03.015

基于计算机虚拟化列车控制系统敏捷测试方法

周庭梁[1,2]，黄　涛[2]，杨文臣[1]，赵时旻[1]

(1. 同济大学 道路与交通工程教育部重点实验室，上海 201804；2. 卡斯柯信号有限公司，上海 510275)

摘要：传统的基于通信的列车运行控制(communication based train control，CBTC)系统测试受真实硬件设备环境的制约而存在瓶颈，针对该问题提出一种基于计算机虚拟化的CBTC系统测试方法.采用计算机虚拟技术模拟物理硬件设备环境，构建CBTC系统虚拟化测试平台.基于该测试平台，设计了一种自适应测试用例约简优排算法，可以在保证测试覆盖率的前提下，约简测试用例的数量，并通过优化测试用例的执行顺序，缩短系统的测试时间.该方法应用于深圳市地铁2号线CBTC系统测试，与传统测试方法进行了对比.实验表明，本测试方法可精简测试用例幅度约45%，测试时间缩短约33%，大幅提高了系统测试效率.

关键词：基于通信的列车运行控制系统；计算机虚拟化；敏捷测试；自适应测试用例约简优排

中图分类号：U284.48；TP311.52　　**文献标志码**：A

Agile Testing for CBTC System Based on Computer Virtualization

ZHOU Tingliang[1,2]，*HUANG Tao*[2]，*YANG Wenchen*[1]，*ZHAO Shimin*[1]

(1. Key Laboratory of Road and Traffic Engineering of the Ministry of Education，Tongji University，Shanghai 201804，China；2. CASCO Signal Co. Ltd，Shanghai 510275，China)

Abstract：A communication based train control (CBTC) system testing method based on computer virtualization was proposed to solve the bottleneck in the traditional system testing due to the constraints of physical hardware environment. The CBTC system virtualization testing platform was built with the computer virtualization technology，which can simulate the real hardware environment. Then an adaptive testing case reduction and optimal scheduler algorithm based on this platform was designed to reduce the number of the testing cases on the premise of guaranteeing the testing coverage，and the system testing time can be decreased through optimizing the execution sequence of the testing cases. For the CBTC system of Shenzhen metro line 2，an experiment was conducted to compare with the traditional testing method. The experiment shows that the proposed method can reduce approximately 45% of the number of testing cases，and decrease approximately 33% of the testing time. So the system testing efficiency is improved.

Key words：communication based train control (CBTC)；computer virtualization；agile test；adaptive testing case minimization and optimal scheduler

基于通信的列车运行控制(communication based train control，CBTC)系统是安全苛求系统，遵循故障导向安全的设计理念，系统安全相关功能须具备SIL(safety integrity level)4——安全完整性最高等级，具有安全高效和易于维护的特点，业已成为现今城市轨道交通领域占主导地位的列车运行控制系统.在CBTC系统全生命周期里，系统集成测试质量决定了系统的研发效果，进而影响工程产品的质量安全水平，是系统安全和性能验证的最直接且最有效手段.

系统集成测试领域典型的测试系统/平台包括RAILSIM(railway analysis and interactive line simulator)系统[1]、ULRAS系统和HIL(hardware in the loop)测试系统[2].北美RAILSIM系统可精确模拟各种铁路系统中的多车追踪运行；日本ULRAS系统可进行列车运行曲线计算，列车模型对运营的影响分析，以及多车运行能力及效果的评价等；意大利HIL系统可模拟实际线路中不可能出现或是可能诱发事故的一些极端测试环境.但是，当前的主流测试系统都依托于被测系统的硬件设备，任何主要硬件设备或接口的更新将致使测试系统进行大范围的改动，且测试流程繁琐，不能满足CBTC系统测试的实时性及高效性需求.

收稿日期：2014-03-18

基金项目：上海张江国家自主创新示范区专项发展资金重大项目(ZJ2012-ZD-011)

第一作者：周庭梁(1980—)，男，博士生，主要研究方向为轨道交通列车运控系统研制、测试及安全评估. E-mail:Leon_ztl@163.com

同时有关统计数据表明,回归测试一般占软件产品测试预算的 80%以上,占软件维护预算的 50%以上[3-4]. 而回归技术的核心是测试用例优化问题. 针对此问题,典型技术有失效用例的识别与修复、测试用例选择、测试用例优先排序(test case prioritization, TCP)等[5-6],还有常见的贪心算法[7-8]、启发式算法[9]、扩张算法[10-11]、整数规划算法[12]以及 CHEN 等人提出的 GE 和 GRE 算法[13-14]. 这些算法都是针对既有测试用例集的优化或简化策略,其测试效果取决于初始化的测试用例集. 为此,章小芳等人提出基于线性搜索的测试需求约简方法(TRR_Linear)[15],王红园等人提出基于需求关键词关联的测试用例约简方法[16],通过约简测试需求集以获得精简的测试需求集,并将约简后的测试需求集作为生成和优化测试用例集的基础,从而实现测试用例集优化. 但这些方法并没有考虑测试用例的执行顺序问题,即如何针对给定的测试用例集进行用例排序,使得测试及早发现系统缺陷.

本文针对 CBTC 测试系统依赖于关键硬件设备和初始测试用例冗余的问题,引入计算机虚拟化理念,提出一种基于计算机虚拟化的 CBTC 系统测试方法. 该方法分析 CBTC 系统的传统测试方法的瓶颈,采用计算机虚拟技术使测试系统与被测系统软件化,提出 CBTC 系统的敏捷测试方法及其自适应测试用例约简优排算法. 最后针对深圳市地铁 2 号线 CBTC 系统测试进行实验,采用两种方法对案例进行系统集成测试,对所提出的虚拟化平台及测试方法进行效用评价.

1　CBTC 系统测试瓶颈分析

CBTC 系统将先进的通信、计算机及控制技术集成应用于列车运行控制,其依托高精度列车定位和连续、高速、双向的数据通信,采用车载和地面安全设备实现移动闭塞,能有效保证列车行车安全及舒适度. CBTC 系统主要由列车自动控制系统(automatic train control , ATC)、列车自动监控系统(automatic train supervision, ATS)、计算机联锁(computer interlocking, CI)、数据通信系统(data communication system, DCS)和维护支持系统(maintenance support system, MSS)五大子系统组成. 其中,ATC 和 CI 主要用于列车运行安全防护,ATS 用于列车运营调度,MSS 用于提供系统状态信息并监控各子系统运行状态,DCS 则负责设备间双向通信.

作为安全苛求系统,CBTC 系统对测试要求苛刻. CBTC 系统最理想的集成测试环境是现场环境,但现场测试系统受限于测试周期长、资金成本高,不利于系统故障模拟,且不可知危险因素多等. 因此,在轨道交通信号控制系统测试领域,目前应用较广泛的是基于真实设备与部分仿真软件相结合的室内测试系统. 使用定制的硬件接口将真实的被测硬件设备接入列车运行仿真系统,通过数据协议的解析和转换,实现仿真模型和实物系统的无差异接入. 但是,这类测试系统存在如下瓶颈:①除测试平台外,仍需要一套真实的 CBTC 系统设备,成本高昂;②占用大面积实验场地,不利于各设备间通信接口的故障注入等测试;③更新被测设备硬件时,需要同步升级软硬件设备的接口程序,测试系统维护困难.

中国城市轨道交通发展正处于建设高峰期. 在地铁项目实施过程中,业主需求的改变以及外部接口系统相关问题的暴露,要求频繁切换被测试的轨道交通线路和存在大量的系统回归测试,CBTC 系统测试压力越来越大,同时留给测试的时间也越来越少. 因此,在确保 CBTC 系统安全测试功能完备的同时,如何减少测试设备、降低测试成本以及缩短测试时间是 CBTC 系统安全性测试验证亟待解决的关键问题.

2　基于计算机虚拟化 CBTC 系统测试平台设计

2.1　CBTC 系统计算机虚拟化

计算机虚拟化是利用高性能计算机建立被仿真系统的模型,并在某些实验条件下对模型进行动态实验的一门综合性技术,具有高效、安全、受环境条件约束少、可改变时间比例尺等优点,已成为复杂系统设计、分析、运行、评价和培训的重要工具. 虚拟化的原理是直接在计算机硬件或主机操作系统上插入一个精简的软件层,该软件层包含一个以动态和透明方式分配硬件资源的虚拟机监视器(即虚拟化管理程序),可使多个操作系统同时运行在单台物理机上,且彼此之间共享硬件资源. 因此,采用虚拟化技术,可在单台物理机上安装并运行多个操作系统和应用程序,每个操作系统和应用程序在其需要时访问相应的资源.

将虚拟化技术应用到 CBTC 系统的测试领域,通过创建与真实设备一样,可运行自身系统和应用

程序的功能齐全的虚拟机,使用虚拟化软件可转变成虚拟化轨道交通信号系统的硬件设备资源.因而,通过在单台物理机上创建多个虚拟机,每个虚拟机只运行一套完整的CBTC子系统,则可摆脱物理硬件设备的限制,实现被测CBTC系统的虚拟化.运行在各虚拟机中的CBTC子系统可在多个环境之间共享同一台物理机的资源,且各子系统间不存在潜在系统冲突.

2.2 基于计算机虚拟化的CBTC测试平台

2.2.1 CBTC系统虚拟化测试平台框架

虚拟化CBTC系统测试平台框架如图1所示.虚拟化CBTC系统测试平台的被测系统都以软件形式运行在服务器的虚拟机中,每个被测系统虚拟机的传输层采用TCP(transmission control protocol)或UDP(user datagram protocol)通信协议与测试平台进行数据交换.同时,为解决测试平台对实时性的需求,使用实时输入输出系统(real-time front end processors,RTFEP)实现对轨旁信号设备模拟状态的实时传输.测试控制台(testing console)的主程序包括通用信号测试平台模块、信号生成模块和测试场景模型,用以实现信号设备的模拟与被测系统之间的数据交互等.通过测试界面与配置管理界面,测试人员可进行列车在线运行、硬件接口功能等的模拟测试,以及测试过程、测试数据、测试用例和测试场景的数据管理及统计分析.

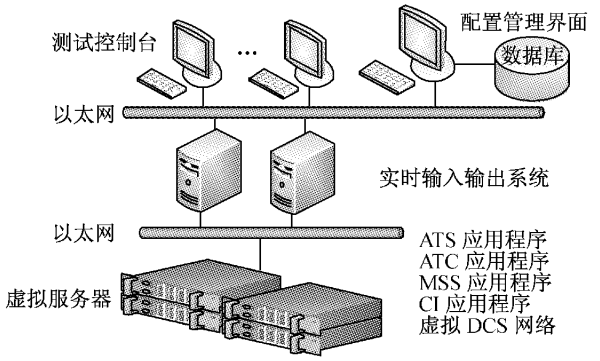

图1 CBTC系统虚拟化测试平台框架

Fig.1 Test platform based on computer virtualization for CBTC system

被测设备ATS,ATC,MSS,CI对应的应用程序分别运行于虚拟服务器的虚拟机中,并采用虚拟DCS网络配置并连接各虚拟机,使所有应用程序之间的通信满足系统测试要求,同时,采用以太网建立被测系统与RTFEP系统的直接通信,则测试控制台通过RTFEP实时系统可直接发送与接收测试命令,而无须处理电信号与网络信号的转换,不必关心特定硬件的接口.因而,整个测试系统可抽象为纯软件之间的交互系统.

2.2.2 CBTC子系统虚拟化配置

在CBTC系统的五大子系统中,ATS,MSS和DCS子系统运行于Windows或Unix操作系统,为此,采用虚拟机管理软件创建与配置相对应的虚拟机,可直接完成ATS,MSS和DCS子系统的部署.但是,ATC和CI子系统存在与外部设备的机电接口,需要模拟通信层以还原其与外部系统的通信过程.以实际车载ATC子系统为例,车载ATC包括ATP(automatic train protection)和ATO(automatic train operation)两个功能模块,分别负责列车自动防护与列车自动驾驶;同时,车载ATC仍包括I/O(input/output)模块,负责采集车辆状态信息并发送给ATP和ATO功能模块进行计算,将相应控制命令发送给车辆执行,并在司机驾驶界面DMI(driver machine interface)上显示,进而通过与轨旁或中心设备网络进行信息交互.为此,I/O模块与车辆存在机电接口,在CBTC系统虚拟化测试平台中,在保证软件功能和线路数据不变的情况下,仍需要模拟车载ATC与外界环境进行信息交互的I/O模块,使运行在虚拟机中的ATC子系统与实际系统的功能完全一致.

车载ATC子系统虚拟主机的配置如图2所示,为使用与真实设备相同的ATP和ATO的软件与数据,增加一个通信层,其负责处理ATP与ATO模块的内部通信,以及ATC与外部设备间的数据交换.既保证了ATP与ATO软件之间的内部数据同步,同时又保证了车载ATC与安全信号网等外部接口的正常通信.通过在虚拟机中部署真实的ATP和ATO模块软件,ATP虚拟执行器与ATO虚拟执行器共同实现车载ATC的全部功能.所有安全数据及非安全数据则可直接使用真实的项目运行数据.

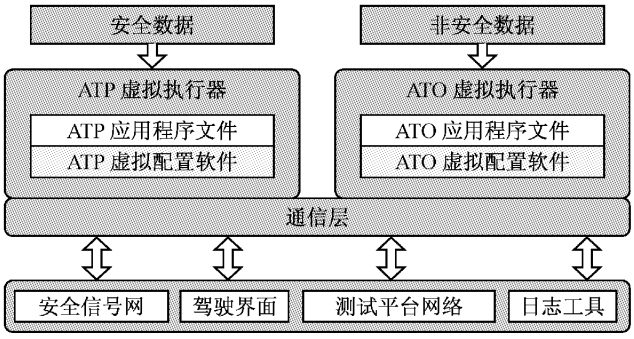

图2 车载ATC子系统虚拟主机配置

Fig.2 Visual host configuration of sub-ATC system

3　虚拟化敏捷测试以及自适应用例约简优排算法

3.1　虚拟化敏捷测试

敏捷测试除测试系统本身外，还包括系统集成过程和模式. 敏捷方法的特点可归纳为：迭代周期短、持续不断的集成、众多团队的交流与沟通以及频繁的反馈[8]. CBTC 系统只是整个城市轨道交通系统的一个组成部分，且在 CBTC 项目实施中，将根据业主需求，分阶段和动态调整系统设计与部署. 因而，CBTC 系统建设过程具备典型敏捷项目实施特点，其测试过程需要不断修正测试指标、及时调整测试策略及确保系统有效需求最终圆满实现等特性，均符合 CBTC 系统建设过程的内在敏捷需求. 为此，结合 CBTC 系统测试的内在特性，依托虚拟化测试平台，引入敏捷测试理论.

虚拟化敏捷测试方法包括三个要素：虚拟化测试平台，敏捷测试方法和测试用例约简优排方法. 虚拟化测试平台采用计算机虚拟技术虚拟 CBTC 系统硬件资源，搭建该软件的系统测试平台. 敏捷测试团队采取的敏捷测试方法除了保证 CBTC 子系统的测试质量，更要保证整个 CBTC 系统集成过程的正确性. 测试用例的设计是 CBTC 敏捷测试的核心，其基准原则是以尽可能少的测试案例发现尽可能多的系统缺陷. 根据项目实施过程中业主对 CBTC 系统需求的频繁更新，测试用例需同步进行动态调整. 因此，本文将重点讨论测试用例约简优排方法.

3.2　测试用例约简优排算法

敏捷测试过程中每一轮测试用例集是根据需求动态变化的，需求的变动会导致用例集的变动. 因此传统测试算法不能从根本上保证满足测试目标，需对测试用例进行最优化. 根据线性搜索测试需求约简方法（TRR_Linear），并针对 Elbaum 等提出的 APFD（average percentage of fault detection）计算公式[17]进行改进，提出一种自适应测试用例约简优排（adaptive test case minimization and optimal scheduler，ATCMOS）算法，通过设定特定排序准则，精简测试用例并优化其执行次序，描述如下.

给定测试需求集 $R=\{r_1, r_2, \cdots, r_m\}$，测试用例集 $T=\{t_1, t_2, \cdots, t_n\}$，测试用例约简旨在寻找 T 的某个子集，该子集中的测试用例可满足测试需求集 R.

定义 1　对于测试需求集 R, R'，若 $|R'|\leqslant|R|$，$\forall r\in R, \exists r'\in R', r'\subseteq r$，且 $\forall r'\in R', \exists r\in R, r'\subseteq r$，则称测试需求集 R' 为 R 的精简测试需求集. 若不存在满足 $|R_s|<|R'|$ 的精简测试需求集 R_s，则称 R' 为 R 的最小精简测试需求集，记为 R''[11].

定义 2　若测试需求集表示为 $R=\{r_1, r_2, \cdots, r_m\}$，测试用例集表示为 $T=\{t_1, t_2, \cdots, t_n\}$.

（1）测试需求 r 到测试用例集 T 的映射 $T(r)=$ {所有可以满足 r 的测试用例 t}；

（2）测试用例集 T 到测试需求 r 的映射 $R(T)=r$，其中：$\forall t\in T$，t 可以满足 r，且 $\forall t\notin T$，t 不能满足 r.

由 $T(r)$ 可以定义测试需求集 R 到测试用例集 T 的映射 $T(R)=\bigcup T(r)$. 若测试需求 r 到测试用例集 T 的映射结果为空集，即 $T(r)=\varnothing$，则测试需求 r 为空，即 $r=\varnothing$；反之亦然.

根据定义 2 可以得出测试需求和测试用例集之间存在着对应关系，即每项测试需求对应着一个非空的测试用例集. 因此，对于测试需求 $r_i, r_j\in R$，$T_i=T(r_i)$，$T_j=T(r_j)$，可以进一步定义测试需求的交、并运算[15]

$$r_i\cap r_j=R(T_i\cap T_j),\ r_i\cup r_j=R(T_i\cup T_j) \tag{1}$$

假设有测试用例集 T，包含 n 个测试用例，m 个缺陷，给定一个测试用例执行次序，其中 T_{fi} 表示首个可检测到第 i 个缺陷的测试用例在该执行次序中所处次序，则 $f(T)$ 的计算公式为

$$f(T)=1-(a_1T_{f1}+a_2T_{f2}+\cdots+a_mT_{fm})/mn+1/2n \tag{2}$$

其中：$\sum_{i=1}^{m}a_i=m$，$0\leqslant a_i\leqslant m$；$a_i$ 代表缺陷的安全等级；由于 T_{fi} 表示首个可检测到第 i 个缺陷的测试用例在该执行次序中所处次序，说明 $a_1T_{f1}+a_2T_{f2}+\cdots+a_mT_{fm}$ 越小，则该用例集次序可以确保越早暴露缺陷，故 f 值越大越优.

定义 3　设 T 的全排列集合为 T_p，排序目标函数 f，最优测试用例 $T'(T'\in T_p)$ 的定义如下：

$$f(T')\geqslant f(T''),\ \forall T''\in T_p, \text{且 } T''\neq T' \tag{3}$$

即在大量的测试用例中筛选出最优测试用例并对其进行优先级排序，得出用最少的测试用例覆盖最多的测试需求，并最早发现最多问题的序列集.

根据上述三个定义，ATCMOS 算法伪代码如下所示：

```
//步骤 1：初始化
R'':=∅;R':=R;
```

//步骤2:约简出最小测试需求集

while (R′不为空) //约简部分重合关系的测试需求

从R′中取出一项测试需求r,R′:=R′-{r};

k:=r;

for each $r_i \in R'(r_i \neq r)$ do

if ($k \cap r_i \neq \varnothing$) then

k:=k∩r_i; //测试需求k为R′中部分测试需求的交集,是约简得到的新的测试需求

R′:=R′-{r_i}; //从R′中删除已被约简的测试需求

end if

end for

R″:=R″+{k}; //将新的测试需求k加入R″

end while

//步骤3:生成最小测试用例集

对R″中每项测试需求的用例映射T(r),根据定义2进行交并运算,得到约简后的T′(r),满足R(T′)=r.然后从T′(r)中为每条需求项挑选一条测试用例,构成优化的测试用例集TS,|TS|=|R″|.

//步骤4:生成最优序列测试用例

根据回归测试所需验证缺陷或功能对应最小测试用例集TS,给每个用例Ti原检测出的缺陷编号为{1,2,3,…,m},对于每个T′中所有元素生成n个序列,分别计算每个序列的f(T);

根据公式(3)得出最大f(T)值的序列.

4 实例分析

4.1 应用实例描述

深圳地铁2号线的CBTC系统及其子系统最先采用的是基于真实设备与部分仿真软件相结合的室内测试系统,且采用逐条需求逐条测试的方法(原测试方法),即为每个需求定义至少一个测试用例.

研究以深圳市地铁2号线CBTC信号系统作为虚拟化测试平台和测试方法的应用对象,采用计算机虚拟化测试方法将2号线的信号系统部署于虚拟化测试平台(虚拟测试方法),将测试对象抽象为纯软件,并以CBTC模式下ATC关于列车驾驶模式转换所对应的测试需求b_1~b_{11}为例,对ATC系统的此功能进行测试用例约简优排研究,如图3所示.

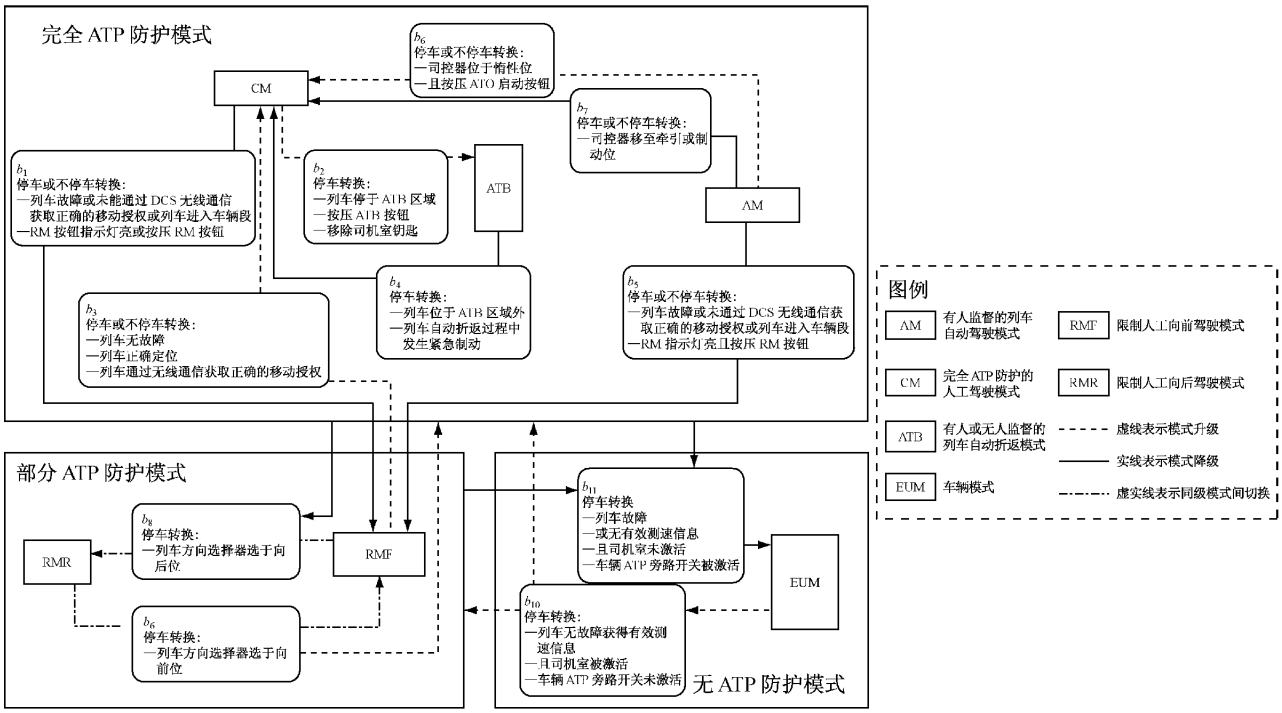

图3 ATC列车驾驶模式转换图

Fig.3 Driving mode conversion of sub-ATC system

4.2 测试用例约简及优排

由图3可知,存在需求集$R=\{b_1, b_2, \cdots, b_{11}\}$,利用ATCMOS算法对$R$进行约简.例如,根据ATC模式转换关系,从AM(有人监督的列车自动驾驶模式)到CM(完全ATP防护的人工驾驶模式)再到RMF(限制人工向前驾驶模式)是一个模式降级过程,故$b_1 \cap b_2 \cap b_4 \cap b_6 = b_6$,由ATCMOS算法步骤2可知,$b_1$,$b_2$和$b_4$作为被约简的测试需求而从$R$中被删除.依此类推,约简后的最小需求集为$R'=\{b_{11} \cap b_5 \cap b_7, b_8, b_6\}$,由于表1中不存在测试用例满足$b_{11} \cap b_5 \cap b_7$,因此,最终的精简测试需求集为$R''=\{b_5 \cap b_7, b_5 \cap b_{11}, b_{11} \cap b_7, b_8, b_6\}$.

表 1　测试用例 $t_1 \sim t_{12}$ 与测试需求 $b_1 \sim b_{11}$ 的满足关系

Tab.1　Constraint relationship between studied test case and test demand

测试用例	测试需求										
	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}
t_1	*	*	*	*	*		*	*	*	*	
t_2	*	*	*	*	*	*		*		*	*
t_3	*	*	*	*		*					*
t_4	*	*	*	*		*					
t_5	*	*	*	*	*			*		*	
t_6	*	*	*	*		*					
t_7	*	*	*	*	*		*	*	*	*	
t_8	*	*	*	*		*					*
t_9	*	*	*	*	*	*		*		*	
t_{10}	*	*	*	*		*	*	*	*	*	*
t_{11}	*	*	*	*	*			*		*	
t_{12}	*	*	*	*	*	*	*	*	*	*	

注：* 代表测试用例满足测试需求.

根据 ATCMOS 算法的步骤 3，对 R'' 中的每项测试需求的用例映射 $T(b)$. 根据定义 2 进行交并运算，得到约简后的 $T'(b_5 \bigcap b_7)=\{t_1,t_7,t_{12}\}$，$T'(b_5 \bigcap b_{11})=\{t_2\}$，$T'(b_{11} \bigcap b_7)=\{t_{10}\}$，$T'(b_6)=\{t_2,t_3,t_4,t_6,t_8,t_9,t_{10},t_{12}\}$，$T'(b_8)=\{t_1,t_2,t_5,t_7,t_9,t_{10},t_{11},t_{12}\}$，从 $T'(b)$ 中为每条需求项挑选一条测试用例，构成优化的测试用例集 T_s，可得出最小测试用例集 $T_s=\{t_1,t_2,t_4,t_5,t_{10}\}$. 这 5 个测试用例对应的缺陷标号如表 2 所示.

表 2　测试用例与缺陷对应实例

Tab.2　Test cases with corresponding defects

测试用例	缺陷标号									
	1	2	3	4	5	6	7	8	9	10
t_1	*				*					
t_2	*	*	*	*	*					
t_4	*				*	*	*			
t_5					*					
t_{10}								*	*	*

注：* 代表测试用例能够检测出的缺陷标号.

默认情况下，所有缺陷的安全等级都一样，此时根据公式(2)，在本例中取值每一个 a_i 都等于 1，即各缺陷安全等级相同，故 $f(T)=1-(T_{f1}+T_{f2}+\cdots+T_{fm})/nm+1/2n$. 以两种测试用例的执行顺序为例，第一种测试用例的执行序列是 t_1-t_2-t_4-t_5-t_{10}，则其 $f(T)$ 为 52%；第二种测试用例的执行序列是 t_{10}-t_5-t_4-t_2-t_1，则其 $f(T)$ 为 58%. 如图 4 所示，$f(T)$ 表示不同测试顺序下，测试用例执行比例与检测出的缺陷比例之间的关系. 例如在测试序列 t_{10}-t_5-t_4-t_2-t_1 下，测试用例 t_{10} 执行后(即执行了 20%的测试用例)，可以检测出 30%的缺陷；测试用例 t_5 执行后，可以检测出 40%的缺陷. 由定义 3 可知，执行序列 t_{10}-t_5-t_4-t_2-t_1 要优于序列 t_1-t_2-t_4-t_5-t_{10}.

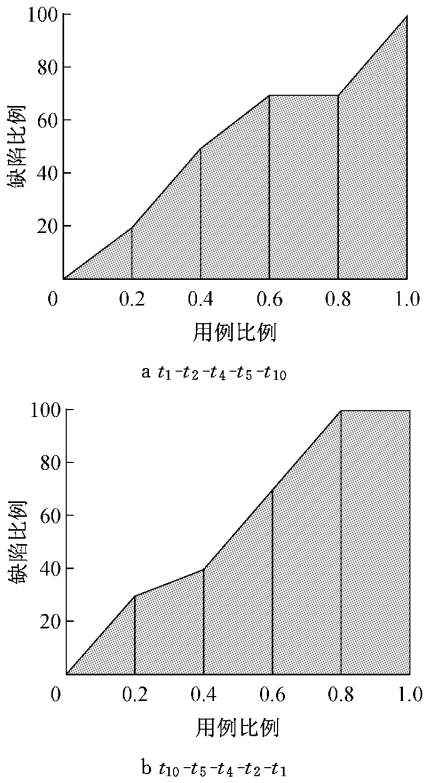

图 4　不同测试顺序的 $f(T)$ 比较分析

Fig.4　$f(T)$ performance comparison of two test sequences

根据 ATCMOS 算法步骤 4，本实例算法最终返回最大 $f(T)$ 的序列 $S=\{t_2,t_{10},t_4,t_5,t_1\}$. 若考虑缺陷不同安全等级，假设从低到高分为 3 类{低，中，高}，对应缺陷安全等级的取值为{0.5，1.0，1.5}.

假设缺陷 6 和 7 的安全等级是高，缺陷 8 和 9 的安全等级是低，其余缺陷的安全等级均为中，此时，存在更优序列{t_2，t_4，t_{10}，t_5，t_1}. 因此，该算法可以根据不同缺陷安全等级动态输出不同测试序列.

4.3　测试算法性能

采用虚拟测试方法对深圳地铁 2 号线的 CBTC 信号系统及其子系统的不同软件版本进行测试，默认所有缺陷安全等级均相同，并与原测试方法比较，以缺陷总数为终止准则，两种测试方法的测试结果如表 3 所示. 可知原测试方法的测试用例执行总数为 2 288，虚拟测试方法的测试用例执行总数为 1 239，缩减幅度约 45%，两种测试方法均能发现相同总数的系统缺陷，但采用虚拟化测试方法所需要的测试用例执行总数明显减少. 原测试方法的工时总数为 1 604 h，虚拟测试方法的工时总数为 1 066 h，相应的测试消耗时间也大幅减小，降低幅度达33%，

且在不同版本的软件系统中,测试方法性能稳定.因而,虚拟化测试方法在保障测试结果精确度的同时,大幅提高了系统测试效率,进而节约测试成本.

表3 深圳地铁2号线信号系统及其子系统的测试结果

Tab.3 Test performance of CBTC system test for Shenzhen metro line 2

子系统/系统名称	发布测试版本数	测试方法	测试用例执行总数	总工时/h	缺陷总数
ATC	3	原测试方法	345	264	24
		虚拟测试方法	135	192	24
ATS	10	原测试方法	860	430	153
		虚拟测试方法	470	310	153
CI	2	原测试方法	452	280	41
		虚拟测试方法	162	220	41
DCS	2	原测试方法	64	150	11
		虚拟测试方法	52	130	11
MSS	5	原测试方法	155	120	18
		虚拟测试方法	140	106	18
CBTC	4	原测试方法	412	360	79
		虚拟测试方法	280	108	79

5 结论

(1)虚拟化CBTC系统测试平台具有CBTC系统硬件虚拟化、软件数据真实化等特性,从而保证测试环境切换的连续性和测试方式的灵活性,使得传统对硬件依赖性强的CBTC系统测试平台演变成为一套高效实用的基于"纯软件控制"的测试平台,有助于降低CBTC系统的测试成本和周期.

(2)ATCMOS算法综合了主流的测试用例优先排序技术以及测试用例约简技术,对测试用例集先约简再对其优化排序,使得算法不受测试用例排列顺序的影响,并以系统输入状态条件间的约束因果关系为准则,约简优排自适应测试用例,大幅缩减测试用例的样本数量,减小系统测试的复杂度.

(3)CBTC系统虚拟化敏捷测试方法以敏捷测试理论为指导,以虚拟化测试平台为支撑,以自适应测试用例约简优排算法为优化,具备良好的缺陷检测能力,测试执行总数和测试工时少,极大地提升了系统的测试效率,能够大幅降低总体拥有成本(TOC)的管理测试资源.

(4)采用计算机虚拟技术完全虚拟化系统测试环境是CBTC系统测试的有效解决方案,研究提出的虚拟化测试平台及方法已经应用于深圳地铁2号线CBTC系统测试中.在未来研究中,将计划应用到北京、上海、广州、深圳等城市地铁CBTC系统测试中,并通过实际运行数据进一步验证所提出方法的效能及稳健性.

参考文献:

[1] Capuder T, Lugaric L, Brekalo S J, *et al*. Optimizing the train power system in Zagreb[C]// Vehicle Power and Propulsion Conference. Dearborn: IEEE, 2009: 41-45.

[2] Addeo F, Allotta B, Malvezzi M, *et al*. ATP/ATC subsystem testing and validation using a HIL test rig[C]// Computers in Railways IX: Computer Aided Design, Manufacture and Operation in the Railway and Other Advanced Mass Transit Systems. Dresden: WIT Press, 2004: 411-420.

[3] Beizer B. Software testing techniques[M]. New York: Van Nostrand Reinhold, 1990.

[4] Leung H, White L. Insights into regression testing[C]// Conference on Software Maintenance. Miami: IEEE, 1989: 60-69.

[5] Yoo S, Harman M. Regression testing minimization, selection and prioritization: A survey[J]. Software Testing, Verification & Reliability, 2012, 22(2): 67.

[6] Harrold M, Orso A. Retesting software during development and maintenance[C]// The Frontiers of Software Maintenance. Beijing: IEEE, 2008: 99-108.

[7] Kichigin D Y. A method of test-suite reduction for regression integration testing[J]. Programming and Computer Software, 2009, 35(5): 282.

[8] Chen T Y, Lau M F. A new heuristic for test suite reduction [J]. Information and Software Technology, 1998, 40(5): 347.

[9] Rout J P, Mishra R, Malu R. An effective test suite reduction using priority cost technique[J]. International Journal of Computer Science & Engineering Technology, 2013, 4(4): 372.

[10] Marre M, Bertolino A. Using spanning sets for coverage testing [J]. IEEE Trans on Software Engineering, 2003, 29(11): 974.

[11] Jeffrey D, Gupta N. Improving fault detection capability by selectively retaining test cases during test suite reduction[J]. IEEE Trans on Software Engineering, 2007, 33(2): 108.

[12] Lee J G, Chung C G. An optimal representative sets election method[J]. Information and Software Technology, 2000, 42 (1): 17.

[13] Xu S P. A methodology for test suite optimization based on testing requirement[C]// The 4th International Conference on Software Engineering and Service Science. Beijing: IEEE, 2013: 216-219.

[14] Chen T Y, Lau M F. On the divide-and-conquer approach towards test suite reduction[J]. Information Sciences, 2003, 152(1): 89.

[15] 章晓芳,徐宝文,聂长海,等.一种基于测试需求约简的测试用例集优化方法[J].软件学报,2007,18(4):821.
ZHANG Xiaofang, XU Baowen, NIE Changhai, *et al*. An approach for optimizing test suite based on testing requirement reduction[J]. Journal of Software, 2007, 18(4): 821.

[16] 王红园,郭永飞,姬琪.面向需求覆盖的航天软件测试用例优化方法[J].光学精密仪器,2014,22(1):228.
WANG Hongyuan, GUO Yongfei, JI Qi. Optimization of aerospace software test cases based on requirement coverage [J]. Optics and Precision Engineering, 2014, 22(1): 228.

[17] Elbaum S, Malishevsky A G, Rothermel G. Prioritizing test cases for regression testing[C]// The International Symposium of Software Testing and Analysis. Portland: ACM Press, 2000: 102-112.