(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 118093352 A
(43)申请公布日 2024.05.28

(21)申请号 202211430311.2

(22)申请日 2022.11.15

(71)申请人 抖音视界有限公司
地址 100041 北京市石景山区实兴大街30号院3号楼2层B-0035房间

(72)发明人 曹紫光 高玉军 毛雪 刘冠成

(74)专利代理机构 上海光栅知识产权代理有限公司 31340
专利代理师 关浩 马雯雯

(51)Int.Cl.
G06F 11/36(2006.01)

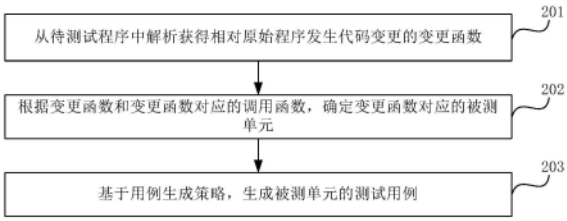
权利要求书4页 说明书18页 附图3页

(54)发明名称

测试用例生成方法、装置、设备、介质及产品

(57)摘要

本公开实施例提供一种测试用例生成方法、装置、设备、介质及产品。测试用例生成方法可以包括:从待测试程序中解析获得相对原始程序发生代码变更的变更函数;根据所述变更函数和所述变更函数对应的调用函数,确定所述变更函数对应的被测单元;所述被测单元为是待测试程序中最小的参与测试的程序模块;基于用例生成策略,生成所述被测单元的测试用例。解决了测试用例生成过低的技术问题。



CN 118093352 A

1.一种测试用例生成方法,其特征在于,包括:

从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

根据所述变更函数和所述变更函数对应的调用函数,确定所述变更函数对应的被测单元;所述被测单元为是待测试程序中最小的参与测试的程序模块;

基于用例生成策略,生成所述被测单元的测试用例。

2.根据权利要求1所述的方法,其特征在于,所述基于用例生成策略,生成所述被测单元的测试用例,包括:

为所述被测单元生成测试参数,获得所述被测单元对应的至少一个测试参数;

基于参数数量最小且对所述被测单元的代码覆盖率最大的参数选择条件,从至少一个所述测试参数中选择目标测试参数;

利用所述目标测试参数,生成所述被测单元的测试用例。

3.根据权利要求2所述的方法,其特征在于,所述为所述被测单元生成测试参数,获得所述被测单元对应的至少一个测试参数,包括:

根据所述变更函数和所述变更函数对应调用函数之间的函数调用关系,确定所述被测单元对应的至少一个分支路径;

为所述分支路径生成至少一个测试参数,以获得至少一个所述分支路径分别对应的至少一个所述测试参数;

确定至少一个所述分支路径分别对应的至少一个所述测试参数为所述被测单元对应的至少一个测试参数。

4.根据权利要求3所述的方法,其特征在于,所述分支路径包括所述变更函数中执行任一次时通过的分支代码和所述分支代码在所述调用函数对应的调用代码;所述为所述分支路径生成至少一个测试参数,包括:

根据所述变更函数的函数信息,确定所述分支路径在所述分支代码对应的第一参数;

基于参数构造器,利用所述函数调用关系中所述分支代码和所述分支代码在所述调用函数对应的调用代码之间的跳转关系,确定所述调用代码对应的第二参数;

确定所述分支代码对应的第一参数和所述调用代码对应的第二参数为所述分支路径的测试参数,返回至根据所述变更函数的函数信息,确定所述分支路径在所述分支代码对应的第一参数的步骤继续执行,直至为所述分支路径生成至少一个测试参数的参数数量满足预设的数量阈值。

5.根据权利要求3所述的方法,其特征在于,所述基于参数数量最小且对所述被测单元的代码覆盖率最大的参数选择条件,从至少一个所述测试参数中选择目标测试参数,包括:

对所述测试参数进行参数测试处理,获得至少一个所述测试参数分别对应的测试结果和代码覆盖信息;

根据至少一个所述测试参数分别对应的测试结果,从至少一个所述测试参数中选择满足测试条件的候选参数;

根据所述候选参数对应的代码覆盖信息,从所述候选参数中选择所述参数选择条件的目标测试参数;所述参数选择条件包括:参数数量最小且对所述被测单元的代码覆盖率最大。

6.根据权利要求5所述的方法,其特征在于,所述对所述测试参数进行参数测试处理,

获得至少一个所述测试参数分别对应的测试结果和代码覆盖信息,包括:

获取为所述被测单元的代码执行覆盖参数插桩处理获得的目标测试单元;

为所述目标测试单元封装程序执行接口;

将所述测试参数输入所述执行程序接口,利用封装于所述程序执行接口内的所述目标测试单元对所述测试参数执行测试处理,获得所述测试参数对应的测试结果和所述代码覆盖信息;

获取至少一个所述测试参数分别输入所述程序执行接口,以获得至少一个所述测试参数分别对应的所述测试结果和所述代码覆盖信息。

7.根据权利要求6所述的方法,其特征在于,所述将所述测试参数输入所述执行程序接口,利用封装于所述程序执行接口内的所述目标测试单元对所述测试参数执行测试处理,获得所述测试参数对应的测试结果和所述代码覆盖信息,包括:

将所述测试参数输入所述执行程序接口,利用封装于所述程序执行接口内的所述目标测试单元对所述测试参数执行测试处理;

若确定所述目标测试单元对所述测试参数执行中断或者编译失败,则确定测试失败为所述测试参数对应的测试结果;

若确定接收到所述目标测试单元对所述测试参数正常返回信息,则获取所述目标测试单元返回的信息为所述测试结果;

根据所述插桩处理的所述覆盖参数进行数据读取,获得所述覆盖参数对应的覆盖数据;

根据所述覆盖参数对应的覆盖数据,生成所述测试参数的所述代码覆盖信息。

8.根据权利要求5所述的方法,其特征在于,所述测试结果包括测试成功或者测试失败;

所述根据至少一个所述测试参数分别对应的测试结果,从至少一个所述测试参数中选择满足测试条件的候选参数,包括:

根据至少一个所述测试参数分别对应的测试结果,从至少一个所述测试参数中选择测试结果为测试成功的正常测试参数;

根据所述被测单元的断言程序对所述正常测试参数的返回信息进行断言测试,获得所述正常测试参数的断言结果;

从所述断言结果中确定断言正常的正常测试参数为所述候选参数。

9.根据权利要求5所述的方法,其特征在于,所述根据所述候选参数对应的代码覆盖信息,从所述候选参数中选择参数数量最小且对所述被测单元的代码覆盖率最大的目标测试参数,包括:

确定以所述参数数量最小且对所述被测单元的代码覆盖率最大作为求解目标的约束求解模型;

将所述候选参数对应的代码覆盖信息输入所述约束求解模型,通过所述约束求解模型求解获得所述候选参数中满足所述求解目标的目标测试参数。

10.根据权利要求2所述的方法,其特征在于,所述利用所述目标测试参数,生成所述被测单元的测试用例,包括:

确定用例生成模板;

将所述目标测试参数插入至所述用例生成模板对应的测试模块,并将所述用例生成模板中的用例参数更新为所述被测单元的单元信息,获得所述被测单元的测试用例。

11.根据权利要求10所述的方法,其特征在于,所述将所述目标测试参数插入至所述用例生成模板对应的测试模块,并将所述用例生成模板中的用例参数更新为所述被测单元的单元信息,获得所述被测单元的测试用例,包括:

将所述目标测试参数插入至所述用例生成模板对应的测试模块,并将所述用例生成模板中的用例参数更新为所述被测单元的单元信息,获得所述被测单元的可执行文件;

基于所述被测单元的可执行文件,建立测试用例的生成消息队列;

响应于线程或进程对所述生成消息队列的消费请求,从所述消息队列中消费所述可执行文件对应的执行进程,获得所述可执行文件执行结束时的所述测试用例。

12.根据权利要求1-11任一项所述的方法,其特征在于,所述待测试程序中的被测单元包括至少一个;所述基于用例生成策略,生成所述被测单元的测试用例之后,还包括:

确定至少一个所述被测单元分别对应的测试用例;

根据至少一个所述被测单元分别对应的文件信息,将属于同一文件信息的所述被测单元划分至同一单元组,获得至少一个单元组;所述单元组包括至少一个所述测试单元;

将所述单元组内的至少一个所述测试单元分别对应的测试用例进行用例合并,获得至少一个文件信息分别对应的文件用例;

确定至少一个所述文件信息分别对应的文件用例为所述待测试程序的目标用例集合。

13.根据权利要求1-11任一项所述的方法,其特征在于,所述从待测试程序中解析获得相对原始程序发生代码变更的变更函数,包括:

响应于针对所述待测试程序的用例生成请求,从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

所述基于用例生成策略,生成所述被测单元的测试用例之后,还包括:

输出为所述被测单元生成的测试用例。

14.根据权利要求1-11任一项所述的方法,其特征在于,所述基于用例生成策略,生成所述被测单元的测试用例之后,还包括:

响应于针对所述待测试程序的程序测试请求,利用所述被测单元的所述测试用例对所述待测试程序进行测试处理,获得测试结果;

输出所述待测试程序的测试结果。

15.根据权利要求1-11任一项所述的方法,其特征在于,还包括:

获取所述待测试程序对应的函式呼叫图;所述函式呼叫图包括所述待测试程序中至少一个函数之间的相互调用关系;

从所述函式呼叫图中查询与所述变更函数存在调用关系的调用函数。

16.一种测试用例生成装置,其特征在于,包括:

程序解析单元,用于从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

调用分析单元,用于根据所述变更函数和所述变更函数对应的调用函数,确定所述变更函数对应的被测单元;所述被测单元为是待测试程序中最小的参与测试的程序模块;

用例生成单元,用于基于用例生成策略,生成所述被测单元的测试用例。

17.一种电子设备,其特征在于,包括:处理器、存储器;

所述存储器存储计算机执行指令;

所述处理器执行所述存储器存储的计算机执行指令,使得所述处理器配置有如权利要求1至15任一项所述的测试用例生成方法。

18.一种计算机可读存储介质,其特征在于,所述计算机可读存储介质中存储有计算机执行指令,当处理器执行所述计算机执行指令时,实现如权利要求1至15任一项所述的测试用例生成方法。

19.一种计算机程序产品,包括计算机程序,其特征在于,所述计算机程序被处理器执行,以配置有如权利要求1至15任一项所述的测试用例生成方法。

测试用例生成方法、装置、设备、介质及产品

技术领域

[0001] 本公开实施例涉及计算机技术领域,尤其涉及一种测试用例生成方法、装置、设备、介质及产品。

背景技术

[0002] 在计算机编程中,单元测试(Unit Testing)也可以称为测试模块,可以是对程序模块进行正确性验证的测试工作。测试单元可以指单个的程序、函数、过程等应用程序中的最小可测试部件,也可称为被测单元。单元测试在实际对程序模块进行测试时,一般是使用测试用例对应用程序中的测试单元的某个执行分支进行测试。但是,一个测试单元可以包括至少一个执行分支,而测试单元的测试结果一般使用代码覆盖率来衡量测试结果的优劣。因此,为了对测试单元中的代码进行更全面的测试,获得更高的代码覆盖率,需要为测试单元生成数量较多的测试用例。

[0003] 目前,测试用例一般是人工来编写的,但是随着应用程序的复杂性的增加,需要编写更多的测试用例来确保被测试的单元的代码覆盖率,导致测试用例的生成效率较低。

发明内容

[0004] 本公开实施例提供一种测试用例生成方法、装置、设备、介质及产品,以克服人工方式编写测试用例导致测试用例生成效率低下的问题。

[0005] 第一方面,本公开实施例提供一种测试用例生成方法,包括:

[0006] 从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

[0007] 根据变更函数和变更函数对应的调用函数,确定变更函数对应的被测单元;被测单元为是待测试程序中最小的参与测试的程序模块;

[0008] 基于用例生成策略,生成被测单元的测试用例。

[0009] 第二方面,本公开实施例提供一种测试用例生成装置,包括:

[0010] 程序解析单元,用于从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

[0011] 调用分析单元,用于根据变更函数和变更函数对应的调用函数,确定变更函数对应的被测单元;被测单元为是待测试程序中最小的参与测试的程序模块;

[0012] 用例生成单元,用于基于用例生成策略,生成被测单元的测试用例。

[0013] 第三方面,本公开实施例提供一种电子设备,包括:处理器以及存储器;

[0014] 存储器存储计算机执行指令;

[0015] 处理器执行存储器存储的计算机执行指令,使得至少一个处理器执行如上第一方面以及第一方面各种可能的设计的测试用例生成方法。

[0016] 第四方面,本公开实施例提供一种计算机可读存储介质,计算机可读存储介质中存储有计算机执行指令,当处理器执行计算机执行指令时,实现如上第一方面以及第一方面各种可能的设计的测试用例生成方法。

[0017] 第五方面,本公开实施例提供一种计算机程序产品,包括计算机程序,计算机程序被处理器执行时实现如上第一方面以及第一方面各种可能的设计的测试用例生成方法。

[0018] 本实施例提供的技术方案,可以从待测试程序中解析获得相对原始程序发生代码变更的变更函数,根据变更函数和变更函数对应的调用函数,确定被测单元。被测单元可以指待测试程序中的最小参与测试的程序模块。通过变更函数可以实现对被测单元的快速定位,利用用例生成策略,生成被测单元的测试用例,提高测试用例的生成效率和准确性。

附图说明

[0019] 为了更清楚地说明本公开实施例或现有技术中的技术方案,下面将对实施例或现有技术描述中所需要使用的附图作一简单地介绍,显而易见地,下面描述中的附图是本公开的一些实施例,对于本领域普通技术人员来讲,在不付出创造性劳动性的前提下,还可以根据这些附图获得其他的附图。

[0020] 图1为本公开实施例提供的一种用于测试用例生成方法的一个应用网络架构图;

[0021] 图2为本公开实施例提供的一种测试用例生成方法的一个实施例的流程图;

[0022] 图3为本公开实施例提供的一种测试用例生成方法的一个实施例的流程图;

[0023] 图4为本公开实施例提供的一种测试用例生成方法的一个实施例的流程图;

[0024] 图5为本公开实施例提供的一种测试用例生成方法的又一个实施例的流程图;

[0025] 图6为本公开实施例提供的一种测试用例生成装置的一个结构示意图;

[0026] 图7为本公开实施例提供的一种电子设备的硬件结构示意图。

具体实施方式

[0027] 为使本公开实施例的目的、技术方案和优点更加清楚,下面将结合本公开实施例中的附图,对本公开实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例是本公开一部分实施例,而不是全部的实施例。基于本公开中的实施例,本领域普通技术人员在没有作出创造性劳动前提下所获得的所有其他实施例,都属于本公开保护的范围。

[0028] 本公开的技术方案可以应用于应用程序的自动化测试场景中,通过对待测试程序进行变更代码的解析,可以实现对变更函数和变更函数对应的调用函数的准确获得,实现对被测单元的准确定位,利用用例生成策略,实现测试用例的自动生成,提高测试用例的生成效率和准确性。

[0029] 相关技术中,测试用例一般采用人工编写,但是这种效率较低。目前,也存在一些用例的自动生成方案,例如基于符号的用例生成方案和基于搜索的用例生成方案。这两种用例的生成方案类似,均可以通过对代码进行统计分析,实现对代码的命中检测来生成相应的测试用例。但是,这种直接对代码进行命中检测分析,以利用代码覆盖率最大作为代码执行所对应的用例集合。这种分析方式较为繁复,需要不断对程序代码进行命中检测,执行复杂度较高,导致用例的生成效率不高。

[0030] 为了解决上述技术问题,本公开中,采用了对待测试程序中发生代码变更的变更函数的检测,以通过变更函数可以对被测单元进行快速定位,利用被测单元和用例生成策略,快速完成测试用例的生成。提高了用例生成效率和准确性。

[0031] 本公开的实施例中,可以从待测试程序中解析获得相对原始程序发生代码变更的

变更函数,根据变更函数和变更函数对应的调用函数,确定被测单元。被测单元可以指待测试程序中的最小参与测试的程序模块。通过变更函数可以实现对被测单元的快速定位,利用用例生成策略,生成被测单元的测试用例,提高测试用例的生成效率和准确性。

[0032] 下面将以具体实施例对本公开的技术方案以及本公开的技术方案如何解决上述技术问题进行详细说明。下面几个具体实施例可以相互结合,对于相同或相似的概念或过程可能在某些实施例中不再赘述。下面将结合附图对本发明的实施例进行详细描述。

[0033] 图1是根据本公开提供的一种用于测试用例生成方法的一个应用网络架构图。根据本公开实施例的应用网络架构中可以包括一个电子设备以及一个与该电子设备通过局域网或者广域网进行网络连接的客户端。假设该电子设备可以为个人计算机、普通服务器,超级个人计算机,云服务器等类型的服务器,本公开中对电子设备的具体类型并不作出过多限定。客户端例如可以为手机、平板电脑、个人计算机、智能家电、可穿戴设备等终端设备,本公中对客户端的具体类型并不作出过多限定。如图2所示,以电子设备为云服务器1,第一客户端2为手机21、平板电脑22为例,任一个客户端2可以向云服务器1提供发起待测试程序的用例生成请求。云服务器1中可以获取基于本公开的技术方案,从待测试程序中解析获得相对原始程序发生代码变更的变更函数,利用变更函数和变更函数确定被测单元,利用用例生成策略,生成被测单元的测试用例,实现测试用例的准确生成。本公开的技术方案中,用例生成策略具体可以实现测试用例的生成,测试用例一般是通过可执行文件生成的,可执行文件可以包括:参数构造器11、断言处理模块12、覆盖率技术器13、用例过滤器14以及用例生成器15。

[0034] 此外,云服务器1可以将生成的测试用例发送至客户端2。客户端2即可以使用查看测试用例,并利用测试用例对待测试程序中的被测单元进行自动化测试。

[0035] 参考图2,图2为本公开实施例提供的一种测试用例生成方法的一个实施例的流程图,该测试用例生成方法可以配置为一测试用例生成装置,测试用例生成装置可以位于电子设备中。测试用例生成方法可以包括以下几个步骤:

[0036] 201:从待测试程序中解析获得相对原始程序发生代码变更的变更函数。

[0037] 原始程序可以为历史版本的应用程序。待测试程序可以为对应用程序进行版本更新,调整程序中的代码所获得的新应用程序,可以对应用程序进行测试。

[0038] 待测试程序可以包括相对更新后的应用程序或者库(Library),待测试程序可以由多个函数组成。本公开所示的函数可以指以编程语言编写的用以自动完成数据处理的程序代码,并不指代数学思想或者数学公式。

[0039] 变更函数可以包括代码发生变更的程序模块,发生变更的代码可以包括一行或多行。

[0040] 可选地,在确定待测试程序之后可以利用代码版本管理程序对待测试程序和历史程序进行代码比较,确定存在代码变更的变更函数。历史程序可以为待测试程序的前一个版本对应的程序代码。历史程序可以为已完成代码测试的程序。

[0041] 202:根据变更函数和变更函数对应的调用函数,确定变更函数对应的被测单元;被测单元为是待测试程序中最小的参与测试的程序模块。

[0042] 变更函数可以为发生代码变更的函数。调用函数可以与变更函数存在调用关系。调用(caller)函数可以是变更函数的所有调用方,调用函数可以被变更函数调用,也即变

更函数可以对调用函数进行调用。当然,在某些实施例中,调用函数也可以调用变更函数,也即,调用函数可以对变更函数进行调用,具体可以根据函数之间的调用和分析需求确定。

[0043] 变更函数和变更函数对应的调用函数可以组成被测单元。参与测试的程序模块可以指用于测试的代码所对应的模块。

[0044] 203:基于用例生成策略,生成被测单元的测试用例。

[0045] 测试用例可以包括用于对被测单元进行测试的测试参数,测试用例可以用于被测单元的测试,将测试用例作为输入数据输入到被测单元可以产生测试结果。被测单元的实际测试可以在被测单元的测试用例生成之后自动执行,以实现自动化的单元测试。

[0046] 本公开实施例中,可以从待测试程序中解析获得相对原始程序发生代码变更的变更函数,根据变更函数和变更函数对应的调用函数,确定被测单元。被测单元可以指待测试程序中的最小的参与测试的程序模块。通过变更函数可以实现对被测单元的快速定位,利用用例生成策略,生成被测单元的测试用例,提高测试用例的生成效率和准确性。

[0047] 为了确保获得的测试用例的准确性更高,可以对测试用例进行筛选。如图3所示,为本公开实施例提供的一种测试用例生成方法的又一个实施例的流程图,该测试用例生成方法可以配置为一测试用例生成装置,测试用例生成装置可以位于电子设备中。与图2所示实施例的不同之处在于,步骤203:基于用例生成策略,生成被测单元的测试用例,可以包括以下几个步骤:

[0048] 301:为被测单元生成测试参数,获得被测单元对应的至少一个测试参数。

[0049] 可以执行一次或多次为被测单元生成测试参数的步骤,以获得被测单元的至少一个测试参数。

[0050] 302:基于参数数量最小且对被测单元的代码覆盖率最大的参数选择条件,从至少一个测试参数中选择目标测试参数。

[0051] 参数数量可以包括目标测试参数的参数数量。参数数量最小是指被选择的目标测试参数的参数数量相较于其他可选择的测试参数的参数数量最小。

[0052] 其他可选择的测试参数可以为代码覆盖率与目标测试参数的代码覆盖率相等,但是参与测试的测试参数的数量要大于目标测试参数的参数数量。

[0053] 被测单元的代码覆盖率可以包括:被测单元的代码中被测试的代码数量和被测单元的所有代码的总代码数量的比值。

[0054] 303:利用目标测试参数,生成被测单元的测试用例。

[0055] 目标测试参数可以包括多个,任一个目标测试参数均可以用于生成被测单元的测试用例。

[0056] 本公开实施例中,在为被测单元生成至少一个测试参数之后,可以通过参数选择条件对至少一个测试参数进行参数筛选,获得目标测试参数。通过参数选择条件的选择可以实现对目标测试参数的准确选择,利用目标测试参数,生成被测单元的测试用例,实现测试用例快速而准确的生成。

[0057] 在实际应用中,至少一个测试参数分别对应的测试用例可以为对测试单元的各个分支路径生成的,实现对测试单元的各个测试路径的准确生成的。如图4所示为本公开实施例提供的一种测试用例生成方法的又一个实施例的流程图,该测试用例生成方法可以配置为一测试用例生成装置,测试用例生成装置可以位于电子设备中。与图3所示实施例的不同

之处在于,步骤301:为被测单元生成测试参数,获得被测单元对应的至少一个测试参数,可以包括以下几个单元:

[0058] 401:根据变更函数和变更函数对应调用函数之间的函数调用关系,确定被测单元对应的至少一个分支路径。

[0059] 在实际应用中,变更函数的可能包含if(如果)语句、switch(选择)语句等用于对同一参数或者结果设置不同参数执行程序的分支程序,同一参数的不同分支程序是并列的。

[0060] 分支路径可以指变更函数中某个分支程序被执行时,该分支程序的调用函数所对应的程序调用路径或者程序跳转路径。一个被测单元可以包括至少一个分支路径。当然,在实际应用中被测单元可以包括一个或多个。

[0061] 402:为分支路径生成至少一个测试参数,以获得至少一个分支路径分别对应的至少一个测试参数。

[0062] 可以为分支路径生成一个或多个测试参数。在实际应用中,由于程序的运行受测试参数的影响,因此,为了确保分支路径被有效测试,可以为分支路径生成至少一个测试参数。每个分支路径均可以对应至少一个测试参数。每个分支路径的测试参数用于测试该分支路径中的代码是否能够顺利执行,并可以利用分支路径的断言程序对执行结果是否准确进行判断。

[0063] 403:确定至少一个分支路径分别对应的至少一个测试参数为被测单元对应的至少一个测试参数。

[0064] 被测单元对应的至少一个测试参数可以包括至少一个分支路径分别对应的至少一个测试参数。

[0065] 本公开实施例中,可以根据变更函数和变更函数对应调用函数之间的函数调用关系,确定被测单元的至少一个分支路径。通过为分支路径生成至少一个测试参数,可以获得至少一个分支路径分别对应的至少一个测试参数,实现至少一个测试参数的准确生成。通过对至少一个分支路径分别生成测试参数,可以确保被测单元中的各个分支路径能够被全面覆盖,确保测试参数的全面性。

[0066] 在一种可能的设计中,分支路径包括变更函数中执行任一次时通过的分支代码和分支代码在调用函数对应的调用代码;为分支路径生成至少一个测试参数,包括:

[0067] 根据变更函数的函数信息,确定分支路径在分支代码对应的第一参数。

[0068] 基于参数构造器,利用函数调用关系中分支代码和分支代码在调用函数对应的调用代码之间的跳转关系,确定调用代码对应的第二参数。

[0069] 确定分支代码对应的第一参数和调用代码对应的第二参数为分支路径的测试参数,返回至根据变更函数的函数信息,确定分支路径在分支代码对应的第一参数的步骤继续执行,直至为分支路径生成至少一个测试参数的参数数量满足预设的数量阈值。

[0070] 其中,变更函数的函数信息可以包括函数签名。函数签名定义了函数或方法的输入与输出,可以包括函数类型、函数名称、函数的参数名称、参数类型、返回参数名称、返回参数类型等信息中的至少一种。分支代码对应的第一参数可以通过变更函数的函数签名直接生成。

[0071] 参数构造器(ArgumentFuzzer)可以基于Fuzz思想构建。在通过函数签名构建分支

代码的第一参数之后,可以利用函数调用关系中分支代码和分支代码在调用函数对应的调用代码之间的跳转关系,通过参数构造器随机构造下游的调用代码的第二参数。

[0072] 本公开实施例中,可以根据变更函数的函数信息,确定分支路径中的分支代码对应的第一参数。此外,还可以利用参数构造器利用函数调用关系,确定调用代码对应的第二参数。通过分别对分支路径中的分支代码和调用代码分别进行参数设置,可以实现更细粒度的参数设置,提高参数设置准确性。

[0073] 为了获得准确的目标测试参数,如图5所示为本公开实施例提供的一种测试用例生成方法的又一个实施例的流程图,该测试用例生成方法可以配置为一测试用例生成装置,测试用例生成装置可以位于电子设备中。与图3所示实施例的不同之处在于,步骤302:基于参数数量最小且对被测单元的代码覆盖率最大的参数选择条件,从至少一个测试参数中选择目标测试参数,可以包括以下几个步骤:

[0074] 501:对测试参数进行参数测试处理,获得至少一个测试参数分别对应的测试结果和代码覆盖信息。

[0075] 可选地,可以对测试参数进行一次代码的预执行,可以将测试参数输入到被测单元的执行程序接口,对测试参数进行测试。

[0076] 测试参数通过执行程序接口获得被测单元的反馈的测试结果。被测单元已设置覆盖参数,因此可以对被测单元的代码覆盖率进行追踪。代码覆盖率具体可以通过覆盖率计数器(Coverage Counter)

[0077] 502:根据至少一个测试参数分别对应的测试结果,从至少一个测试参数中选择满足测试条件的候选参数。

[0078] 测试条件可以包括测试结果正常。

[0079] 503:根据候选参数对应的代码覆盖信息,从候选参数中选择参数选择条件的目标测试参数;参数选择条件包括:参数数量最小且对被测单元的代码覆盖率最大。

[0080] 其中,被测单元的代码覆盖率最大可以指被测单元的代码覆盖为100%,当代码覆盖率对应被测单元进行覆盖测试。

[0081] 在一种可能的设计中,步骤502和步骤503可以由用例过滤器完成。通过用例过滤器可以利用测试条件对候选参数进行初步筛选,获得合格的候选参数,可以利用参数选择条件对候选参数进行更详细筛选,以确保选择的目标测试参数能够达到数量最小且覆盖率最大。

[0082] 本公开实施例中,可以对测试参数进行参数测试处理,获得至少一个测试参数分别对应的测试结果和代码覆盖信息。通过根据至少一个测试参数分别对应的测试结果,可以从至少一个测试参数中选择满足测试条件的候选参数。候选参数的选择即是利用测试条件对至少一个测试条件的初步选择。通过候选参数对应的代码覆盖信息,可以从候选参数中选择参数数量最小且代码覆盖率最大的目标测试参数,实现对参数的二次选择。通过参数的二次选择可以对候选参数进行更详细的参数选择,提高参数选择效率和准确性。

[0083] 在某些实施例中,对测试参数进行参数测试处理,获得至少一个测试参数分别对应的测试结果和代码覆盖信息,包括:

[0084] 获取为被测单元的代码执行覆盖参数插桩处理获得的目标测试单元。

[0085] 为目标测试单元封装程序执行接口。

[0086] 将测试参数输入执行程序接口,利用封装于程序执行接口内的目标测试单元对测试参数执行测试处理,获得测试参数对应的测试结果和代码覆盖信息。

[0087] 获取至少一个测试参数分别输入程序执行接口,以获得至少一个测试参数分别对应的测试结果和代码覆盖信息。

[0088] 可以将被测单元的代码中增加覆盖参数语句,完成覆盖参数的插桩,获得目标测试单元。可以通过覆盖率计数器对测试参数在目标测试单元进行覆盖就的读取,获得测试参数对应的代码覆盖信息。

[0089] 本公开实施例中,通过插桩处理完成对被测单元的覆盖统计,通过为目标测试单元封装程序执行接口,完成目标测试单元的可执行程序的生成,使得目标测试单元能够被执行。通过执行程序接口可以对目标测试单元进行测试并通过覆盖参数获取相应的代码覆盖信息,实现对目标测试单元的插桩和测试,完成被测试单元的自动运行设置,确保后续方案的顺利运行。

[0090] 在一种可能的设计中,将测试参数输入执行程序接口,利用封装于程序执行接口内的目标测试单元对测试参数执行测试处理,获得测试参数对应的测试结果和代码覆盖信息,包括:

[0091] 将测试参数输入执行程序接口,利用封装于程序执行接口内的目标测试单元对测试参数执行测试处理;

[0092] 若确定目标测试单元对测试参数执行中断或者编译失败,则确定测试失败为测试参数对应的测试结果;

[0093] 若确定接收到目标测试单元对测试参数正常返回信息,则获取目标测试单元返回的信息为测试结果;

[0094] 根据插桩处理的覆盖参数进行数据读取,获得覆盖参数对应的覆盖数据;

[0095] 根据覆盖参数对应的覆盖数据,生成测试参数的代码覆盖信息。

[0096] 在将测试参数输入程序执行接口之后,可以利用封装于程序执行接口内的目标测试单元对测试参数执行测试处理,获得相应的处理结果。可以利用用例过滤器(CasesFilter)对测试参数进行过滤。其中,在目标测试单元对测试参数执行的处理结果为参数执行中断或者编译失败,则确定测试异常为测试结果。若处理结果为正常处理,并获得正常的返回信息,则可以获取目标测试单元返回的正常返回信息作为测试结果。之后,用例过滤器可以进行用例筛选,也即获得测试参数的代码覆盖信息,利用测试参数的代码覆盖信息,和约束求解模型对测试参数进行进一步筛选,以获得用例数量最小且覆盖率最高的目标测试参数。

[0097] 本公开实施例中,可以在测试参数输入到执行程序接口之后,可以将程序执行接口内的目标测试单元对测试参数执行测试处理,可以对该测试参数执行测试处理的过程进行检测,以及时获得目标测试单元的测试结果,提高测试结果的获取时效性。另外,通过对插桩处理的覆盖参数进行数据读取,可以对测试参数的代码覆盖信息进行更精确的获取,利用插桩方式可以对测试参数的代码覆盖情况准确获取,提高获取效率和准确度。

[0098] 在某些实施例中,测试结果包括测试成功或者测试失败;

[0099] 根据至少一个测试参数分别对应的测试结果,从至少一个测试参数中选择满足测试条件的候选参数,包括:

[0100] 根据至少一个测试参数分别对应的测试结果,从至少一个测试参数中选择测试结果为测试成功的正常测试参数;

[0101] 根据被测单元的断言程序对正常测试参数的返回信息进行断言测试,获得正常测试参数的断言结果;

[0102] 从断言结果中确定断言正常的正常测试参数为候选参数。

[0103] 可选地,断言程序可以实时生成也可以历史生成。通过断言程序对测试参数进行断言处理,实现对测试参数输入后反馈的信息是否准确进行了判断。

[0104] 断言程序的生成可以是通过对多个测试结果进行字段获取和命中率确定获得,具体可以参考相关技术的描述,在此不再赘述。

[0105] 断言结果可以包括断言正常和断言异常。断言正常的正常测试参数可以作为候选参数。也即是候选参数可以是被测试成功且断言测试正常的测试参数。

[0106] 本公开实施例中,可以对至少一个测试参数分别对应的测试结果中测试成功的正常测试参数进行获取,对正常测试参数的返回信息进行断言,以判断正常测试参数的返回信息是否异常,从断言结果中确定断言结果正常的正常测试参数为候选参数。通过从测试结果是否正常和测试结果中的返回信息是否正常两个递进式的检测判断,可以完成对候选参数的精准选择。

[0107] 为了获得更准确的目标测试参数,根据候选参数对应的代码覆盖信息,从候选参数中选择参数数量最小且对被测单元的代码覆盖率最大的目标测试参数,可以包括:

[0108] 确定以参数数量最小且对被测单元的代码覆盖率最大作为求解目标的约束求解模型;

[0109] 将候选参数对应的代码覆盖信息输入约束求解模型,通过约束求解模型求解获得候选参数中满足求解目标的目标测试参数。

[0110] 可选地,用例过滤器中可以包括约束求解模型。通过测试结果对候选参数初步筛选之后,可以利用用例过滤器中的约束求解模型对候选参数进行进行一步筛选。也即,利用约束求解模型对候选参数在参数选择条件的约束下求交集,以交集计算结果为能够包含所有代码且参数数量最小作为求解目标,获得目标测试参数。

[0111] 可选地,代码覆盖信息可以包括代码覆盖向量。候选参数对应的代码覆盖向量可以通过候选参数所覆盖的分支路径和未覆盖的分支路径确定。以目标测试单元包含5个分支路径为例,某个分支路径被测试,其在候选参数的代码覆盖向量中取值为1,分支路径未为被测,其在候选参数的代码覆盖向量中取值为0。假设获得的候选参数使用代码覆盖向量表示时,可以包括:(0 0 0 0 1)(1 0 0 0 1)(0 1 1 1 1)(0 1 0 0 0),约束求解模型可以从中选择数量最小且覆盖率最高的目标测试参数,覆盖率最高可以包括:所有分支路径被全部覆盖。也即,约束求解模型可以对代码覆盖向量进行交集计算,并通过交集计算查找满足约束条件的参数集合。上述代码覆盖向量中(1 0 0 0 1)和(0 1 1 1 1)两个向量求交集获得(1 1 1 1 1),也即,通过这两个代码覆盖向量各自的候选参数可以作为能够获得能够覆盖全部分支路径,且数量最小的目标测试参数。

[0112] 本公开实施例中,可以通过约束求解模型对目标测试参数进行求解,获得准确的目标测试参数。

[0113] 在一种可能的设计中,为了快速完成测试用例的生成,利用目标测试参数,生成被

测单元的测试用例,可以包括:

[0114] 确定用例生成模板;

[0115] 将目标测试参数插入至用例生成模板对应的测试模块,并将用例生成模板中的用例参数更新为被测单元的单元信息,获得被测单元的测试用例。

[0116] 用例生成模板可以是根据测试环境、测试语言程序应用类型或者场景等信息预先编译好的程序。可以将目标测试参数转换为与用例生成模板相一致的测试模块。

[0117] 本公开实施例中,通过用例生成模板将目标测试参数插入为在用例生成模板对应的测试模块,并将用例生成模板中的用例参数更新为被测单元的单元信息,实现对被测单元的测试用例的准确生成。

[0118] 其中,将目标测试参数插入至用例生成模板对应的测试模块,并将用例生成模板中的用例参数更新为被测单元的单元信息,获得被测单元的测试用例,包括:

[0119] 将目标测试参数插入至用例生成模板对应的测试模块,并将用例生成模板中的用例参数更新为被测单元的单元信息,获得被测单元的可执行文件;

[0120] 基于被测单元的可执行文件,建立测试用例的生成消息队列;

[0121] 响应于线程或进程对生成消息队列的消费请求,从消息队列中消费可执行文件对应的执行进程,获得可执行文件执行结束时的测试用例。

[0122] 用例的生成方法可以将测试用例编译成可执行文件,可执行文件被打包生成消息队列,并将消息队列进行消费执行,以获得测试用例。可执行文件未被执行时是一段可执行的代码,需要触发对可执行文件的执行,通过消息队列的方式可以实现可执行文件的消费执行,实现测试用例的自动生成。

[0123] 基于被测单元的可执行文件,建立测试用例的生成消息队列,可以包括:将被测单元的可执行文件进行打包处理,将打包处理后的可执行文件输入至消息队列。

[0124] 本公开实施例中,通过消息队列方式快速完成测试用例的生成。

[0125] 在一种可能的设计中,待测试程序中的被测单元包括至少一个;基于用例生成策略,生成被测单元的测试用例之后,还包括:

[0126] 确定至少一个被测单元分别对应的测试用例;

[0127] 根据至少一个被测单元分别对应的文件信息,将属于同一文件信息的被测单元划分至至同一单元组,获得至少一个单元组;单元组包括至少一个测试单元;

[0128] 将单元组内的至少一个测试单元分别对应的测试用例进行用例合并,获得至少一个文件信息分别对应的文件用例;

[0129] 确定至少一个文件信息分别对应的文件用例为待测试程序的目标用例集合。

[0130] 可以为每个测试单元关联文件信息。将单元组内得至少一个测试单元分别对应的测试用例进行用例合并,可以包括:将单元组内的至少一个测试单元分别对应的测试用例进行测试代码的拼接和合并,获得该单元组对应文件的文件用例。将至少一个测试单元分别对应的测试用例进行测试代码的拼接可以指:按照各个测试用例所对应的测试单元在文件的位置和调用关系,将至少一个测试用例分别对应的测试代码进行拼接。

[0131] 本公开实施例中,通过将同一文件信息的测试单元的测试用例进行合并,可以以文件作为被测试的对象,一次文件测试可以同时完成该文件下的多个测试单元的测试,提高测试效率和整体性。

[0132] 在一种可能的设计中,从待测试程序中解析获得相对原始程序发生代码变更的变更函数,包括:

[0133] 响应于针对待测试程序的用例生成请求,从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

[0134] 基于用例生成策略,生成被测单元的测试用例之后,还包括:

[0135] 输出为被测单元生成的测试用例。

[0136] 输出为被测单元生成的测试用例可以包括将测试用户发送至客户端,由客户端显示该测试用例。还可以包括:控制显示装置直接显示测试用例。

[0137] 测试用例的输出可以使得用户查看相应的测试用例,因此,在实际应用中,用户还可以通过客户端对测试用例进行更新。电子设备可以检测用户对测试用例执行的更新操作,利用更新操作对应的更新内容更新测试用例,获得最新的测试用例。

[0138] 本公开实施例中,对用例测试请求进行响应,实现测试用例的生成,并输出该测试用例,实现测试用例的生成交互,提高信息处理效率。

[0139] 在一种可能的设计中,基于用例生成策略,生成被测单元的测试用例之后,还包括:

[0140] 响应于针对待测试程序的程序测试请求,利用被测单元的测试用例对待测试程序进行测试处理,获得测试结果;

[0141] 输出待测试程序的测试结果。

[0142] 其中,程序测试请求可以由客户端发送,客户端可以检测用户触发的程序测试请求,并将程序测试请求发送至电子设备。程序测试请求还可以由电子设备自动触发生成,例如在确定测试用例生成结束时,可以自动触发生成待测试程序的程序测试请求。

[0143] 本公开实施例中,在获得测试用例之后,可以在接收到程序测试请求之后,利用被测单元的测试用例对待测试程序进行测试处理,获得测试结果。通过被测单元的自动化测试可以提高被测试单元的测试效率。

[0144] 在某些实施例中,该方法还可以包括:

[0145] 获取待测试程序对应的函式呼叫图;函式呼叫图包括待测试程序中至少一个函数之间的相互调用关系;

[0146] 从函式呼叫图中查询与变更函数存在调用关系的调用函数。

[0147] 可选地,函式呼叫图(callgraph)可以通过将待测试程序输入AST(Abstract Syntax Tree,抽象语法树)解析获得。抽象语法树可以以树的结构解析出源码的数据结构,树上的节点可以表示源码的一种结构,该结构可以包括函数。树上节点之间的边可以指两个结构之间的调用关系。

[0148] 函式呼叫图可以实时生成也可以预先生成。获取待测试程序对应的函式呼叫图可以包括:将待测试程序输入AST解析获得函式呼叫图。获取待测试程序对应的函式呼叫图还可以包括:读取待测试程序的函式呼叫图文件,解析该文件获得函式呼叫图。

[0149] 本公开实施例中,利用函式呼叫图对至少一个函数之间的相互调用关系进行获取,实现调用函数的快速而准确的查询。

[0150] 如图6所示,为本公开实施例提供的一种测试用例生成装置的一个实施例的结构示意图,该测试用例生成装置可以位于电子设备中。其中,测试用例生成装置600可以包括

以下几个单元：

[0151] 程序解析单元601：用于从待测试程序中解析获得相对原始程序发生代码变更的变更函数。

[0152] 调用分析单元602：用于根据变更函数和变更函数对应的调用函数，确定变更函数对应的被测单元；被测单元为是待测试程序中最小的参与测试的程序模块。

[0153] 用例生成单元603：用于基于用例生成策略，生成被测单元的测试用例。

[0154] 作为一个实施例，用例生成单元，包括：

[0155] 参数确定模块，用于为被测单元生成测试参数，获得被测单元对应的至少一个测试参数；

[0156] 参数选择模块，用于基于参数数量最小且对被测单元的代码覆盖率最大的参数选择条件，从至少一个测试参数中选择目标测试参数；

[0157] 用例生成模块，用于利用目标测试参数，生成被测单元的测试用例。

[0158] 在一种可能的设计中，参数确定模块，可以包括：

[0159] 分支确定子模块，用于根据变更函数和变更函数对应调用函数之间的函数调用关系，确定被测单元对应的至少一个分支路径；

[0160] 参数生成子模块，用于为分支路径生成至少一个测试参数，以获得至少一个分支路径分别对应的至少一个测试参数；

[0161] 参数确定子模块，用于确定至少一个分支路径分别对应的至少一个测试参数为被测单元对应的至少一个测试参数。

[0162] 在一种可能的设计中，分支路径包括变更函数中执行任一次时通过的分支代码和分支代码在调用函数对应的调用代码；参数生成子模块具体可以用于：

[0163] 根据变更函数的函数信息，确定分支路径在分支代码对应的第一参数；

[0164] 基于参数构造器，利用函数调用关系中分支代码和分支代码在调用函数对应的调用代码之间的跳转关系，确定调用代码对应的第二参数；

[0165] 确定分支代码对应的第一参数和调用代码对应的第二参数为分支路径的测试参数，返回至根据变更函数的函数信息，确定分支路径在分支代码对应的第一参数的步骤继续执行，直至为分支路径生成至少一个测试参数的参数数量满足预设的数量阈值。

[0166] 在一种可能的设计中，参数选择模块，包括：

[0167] 参数测试子模块，用于对测试参数进行参数测试处理，获得至少一个测试参数分别对应的测试结果和代码覆盖信息；

[0168] 候选选择子模块，用于根据至少一个测试参数分别对应的测试结果，从至少一个测试参数中选择满足测试条件的候选参数；

[0169] 目标选择子模块，用于根据候选参数对应的代码覆盖信息，从候选参数中选择参数选择条件的目标测试参数；参数选择条件包括：参数数量最小且对被测单元的代码覆盖率最大。

[0170] 在一种可能的设计中，参数测试子模块，具体可以用于：

[0171] 获取为被测单元的代码执行覆盖参数插桩处理获得的目标测试单元；

[0172] 为目标测试单元封装程序执行接口；

[0173] 将测试参数输入执行程序接口，利用封装于程序执行接口内的目标测试单元对测

试参数执行测试处理,获得测试参数对应的测试结果和代码覆盖信息;

[0174] 获取至少一个测试参数分别输入程序执行接口,以获得至少一个测试参数分别对应的测试结果和代码覆盖信息。

[0175] 在某些实施例中,参数测试子模块,具体还可以用于:

[0176] 将测试参数输入执行程序接口,利用封装于程序执行接口内的目标测试单元对测试参数执行测试处理;

[0177] 若确定目标测试单元对测试参数执行中断或者编译失败,则确定测试失败为测试参数对应的测试结果;

[0178] 若确定接收到目标测试单元对测试参数正常返回信息,则获取目标测试单元返回的信息为测试结果;

[0179] 根据插桩处理的覆盖参数进行数据读取,获得覆盖参数对应的覆盖数据;

[0180] 根据覆盖参数对应的覆盖数据,生成测试参数的代码覆盖信息。

[0181] 作为一个实施例,测试结果包括测试成功或者测试失败;

[0182] 候选选择子模块,包括:

[0183] 正常选择子模块,用于根据至少一个测试参数分别对应的测试结果,从至少一个测试参数中选择测试结果为测试成功的正常测试参数;

[0184] 断言测试子模块,用于根据被测单元的断言程序对正常测试参数的返回信息进行断言测试,获得正常测试参数的断言结果;

[0185] 从断言结果中确定断言正常的正常测试参数为候选参数。

[0186] 在某些实施例中,目标选择子模块具体可以用于:

[0187] 确定以参数数量最小且对被测单元的代码覆盖率最大作为求解目标的约束求解模型;

[0188] 将候选参数对应的代码覆盖信息输入约束求解模型,通过约束求解模型求解获得候选参数中满足求解目标的目标测试参数。

[0189] 作为一个实施例,用例生成模块可以包括:

[0190] 模板确定子模块,用于确定用例生成模板;

[0191] 参数插入子模块,用于将目标测试参数插入至用例生成模板对应的测试模块,并将用例生成模板中的用例参数更新为被测单元的单元信息,获得被测单元的测试用例。

[0192] 在一种可能的设计中,参数插入子模块,具体用于:

[0193] 将目标测试参数插入至用例生成模板对应的测试模块,并将用例生成模板中的用例参数更新为被测单元的单元信息,获得被测单元的可执行文件;

[0194] 基于被测单元的可执行文件,建立测试用例的生成消息队列;

[0195] 响应于线程或进程对生成消息队列的消费请求,从消息队列中消费可执行文件对应的执行进程,获得可执行文件执行结束时的测试用例。

[0196] 作为一个实施例,待测试程序中的被测单元包括至少一个;该装置还包括:

[0197] 用例确定单元,用于确定至少一个被测单元分别对应的测试用例;

[0198] 文件划分单元,用于根据至少一个被测单元分别对应的文件信息,将属于同一文件信息的被测单元划分至至同一单元组,获得至少一个单元组;单元组包括至少一个测试单元;

[0199] 用例合并单元,用于将单元组内的至少一个测试单元分别对应的测试用例进行用例合并,获得至少一个文件信息分别对应的文件用例;

[0200] 集合确定单元,用于确定至少一个文件信息分别对应的文件用例为待测试程序的目标用例集合。

[0201] 在某些实施例中,程序解析单元,包括:

[0202] 用例响应模块,用于响应于针对待测试程序的用例生成请求,从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

[0203] 该装置还可以包括:

[0204] 第一输出单元,用于输出为被测单元生成的测试用例。

[0205] 作为又一个实施例,还包括:

[0206] 测试响应单元,用于响应于针对待测试程序的程序测试请求,利用被测单元的测试用例对待测试程序进行测试处理,获得测试结果;

[0207] 第二输出单元,用于输出待测试程序的测试结果。

[0208] 在某些实施例中,还包括:

[0209] 函式确定单元,用于获取待测试程序对应的函式呼叫图;函式呼叫图包括待测试程序中至少一个函数之间的相互调用关系;

[0210] 关系查询单元,用于从函式呼叫图中查询与变更函数存在调用关系的调用函数。

[0211] 本实施例提供的装置,可用于执行上述方法实施例的技术方案,其实现原理和技术效果类似,本实施例此处不再赘述。

[0212] 为了实现上述实施例,本公开实施例还提供了一种电子设备。

[0213] 参考图7,其示出了适于用来实现本公开实施例的电子设备700的结构示意图,该电子设备700可以为终端设备或服务器。其中,终端设备可以包括但不限于诸如移动电话、笔记本电脑、数字广播接收器、个人数字助理(Personal Digital Assistant,简称PDA)、平板电脑(Portable Android Device,简称PAD)、便携式多媒体播放器(Portable Media Player,简称PMP)、车载终端(例如车载导航终端)等等的移动终端以及诸如数字TV、台式计算机等等的固定终端。图7示出的电子设备仅仅是一个示例,不应对本公开实施例的功能和使用范围带来任何限制。

[0214] 如图7所示,电子设备700可以包括处理装置(例如中央处理器、图形处理器等)701,其可以根据存储在只读存储器(Read Only Memory,简称ROM)702中的程序或者从存储装置708加载到随机访问存储器(Random Access Memory,简称RAM)703中的程序而执行各种适当的动作和处理。在RAM 703中,还存储有电子设备700操作所需的各种程序和数据。处理装置701、ROM702以及RAM 703通过总线704彼此相连。输入/输出(I/O)接口705也连接至总线704。

[0215] 通常,以下装置可以连接至I/O接口705:包括例如触摸屏、触摸板、键盘、鼠标、摄像头、麦克风、加速度计、陀螺仪等的输入装置706;包括例如液晶显示器(Liquid Crystal Display,简称LCD)、扬声器、振动器等的输出装置707;包括例如磁带、硬盘等的存储装置708;以及通信装置709。通信装置709可以允许电子设备700与其他设备进行无线或有线通信以交换数据。虽然图7示出了具有各种装置的电子设备700,但是应理解的是,并不要求实施或具备所有示出的装置。可以替代地实施或具备更多或更少的装置。

[0216] 特别地,根据本公开的实施例,上文参考流程图描述的过程可以被实现为计算机软件程序。例如,本公开的实施例包括一种计算机程序产品,其包括承载在计算机可读介质上的计算机程序,该计算机程序包含用于执行流程图所示的方法的程序代码。在这样的实施例中,该计算机程序可以通过通信装置709从网络上被下载和安装,或者从存储装置708被安装,或者从ROM 702被安装。在该计算机程序被处理装置701执行时,执行本公开实施例的方法中限定的上述功能。

[0217] 需要说明的是,本公开上述的计算机可读介质可以是计算机可读信号介质或者计算机可读存储介质或者是上述两者的任意组合。计算机可读存储介质例如可以是——但不限于——电、磁、光、电磁、红外线、或半导体的系统、装置或器件,或者任意以上的组合。计算机可读存储介质的更具体的例子可以包括但不限于:具有一个或多个导线的电连接、便携式计算机磁盘、硬盘、随机访问存储器(RAM)、只读存储器(ROM)、可擦式可编程只读存储器(EPROM或闪存)、光纤、便携式紧凑磁盘只读存储器(CD-ROM)、光存储器件、磁存储器件、或者上述的任意合适的组合。在本公开中,计算机可读存储介质可以是任何包含或存储程序的有形介质,该程序可以被指令执行系统、装置或者器件使用或者与其结合使用。而在本公开中,计算机可读信号介质可以包括在基带中或者作为载波一部分传播的数据信号,其中承载了计算机可读的程序代码。这种传播的数据信号可以采用多种形式,包括但不限于电磁信号、光信号或上述的任意合适的组合。计算机可读信号介质还可以是计算机可读存储介质以外的任何计算机可读介质,该计算机可读信号介质可以发送、传播或者传输用于由指令执行系统、装置或者器件使用或者与其结合使用的程序。计算机可读介质上包含的程序代码可以用任何适当的介质传输,包括但不限于:电线、光缆、RF(射频)等等,或者上述的任意合适的组合。

[0218] 上述计算机可读介质可以是上述电子设备中所包含的;也可以是单独存在,而未装配入该电子设备中。

[0219] 上述计算机可读介质承载有一个或者多个程序,当上述一个或者多个程序被该电子设备执行时,使得该电子设备执行上述实施例所示的方法。

[0220] 可以以一种或多种程序设计语言或其组合来编写用于执行本公开的操作的计算机程序代码,上述程序设计语言包括面向对象的程序设计语言—诸如Java、Smalltalk、C++,还包括常规的过程式程序设计语言—诸如“C”语言或类似的程序设计语言。程序代码可以完全地在用户计算机上执行、部分地在用户计算机上执行、作为一个独立的软件包执行、部分在用户计算机上部分在远程计算机上执行、或者完全在远程计算机或服务器上执行。在涉及远程计算机的情形中,远程计算机可以通过任意种类的网络——包括局域网(Local Area Network,简称LAN)或广域网(Wide Area Network,简称WAN)—连接到用户计算机,或者,可以连接到外部计算机(例如利用因特网服务提供商来通过因特网连接)。

[0221] 附图中的流程图和框图,图示了按照本公开各种实施例的系统、方法和计算机程序产品的可能实现的体系架构、功能和操作。在这点上,流程图或框图中的每个方框可以代表一个模块、程序段、或代码的一部分,该模块、程序段、或代码的一部分包含一个或多个用于实现规定的逻辑功能的可执行指令。也应当注意,在有些作为替换的实现中,方框中所标注的功能也可以以不同于附图中所标注的顺序发生。例如,两个接连地表示的方框实际上可以基本并行地执行,它们有时也可以按相反的顺序执行,这依所涉及的功能而定。也要注

意的是,框图和/或流程图中的每个方框、以及框图和/或流程图中的方框的组合,可以用执行规定的功能或操作的专用的基于硬件的系统来实现,或者可以用专用硬件与计算机指令的组合来实现。

[0222] 描述于本公开实施例中所涉及到的单元可以通过软件的方式实现,也可以通过硬件的方式来实现。其中,单元的名称在某种情况下并不构成对该单元本身的限定,例如,第一获取单元还可以被描述为“获取至少两个网际协议地址的单元”。

[0223] 本文中以上描述的功能可以至少部分地由一个或多个硬件逻辑部件来执行。例如,非限制性地,可以使用的示范类型的硬件逻辑部件包括:现场可编程门阵列(FPGA)、专用集成电路(ASIC)、专用标准产品(ASSP)、片上系统(SOC)、复杂可编程逻辑设备(CPLD)等等。

[0224] 在本公开的上下文中,机器可读介质可以是有形的介质,其可以包含或存储以供指令执行系统、装置或设备使用或与指令执行系统、装置或设备结合地使用的程序。机器可读介质可以是机器可读信号介质或机器可读储存介质。机器可读介质可以包括但不限于电子的、磁性的、光学的、电磁的、红外的、或半导体系统、装置或设备,或者上述内容的任何合适组合。机器可读存储介质的更具体示例会包括基于一个或多个线的电气连接、便携式计算机盘、硬盘、随机存取存储器(RAM)、只读存储器(ROM)、可擦除可编程只读存储器(EPROM或快闪存储器)、光纤、便捷式紧凑盘只读存储器(CD-ROM)、光学储存设备、磁储存设备、或上述内容的任何合适组合。

[0225] 第一方面,根据本公开的一个或多个实施例,提供了一种测试用例生成方法,包括:

[0226] 从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

[0227] 根据变更函数和变更函数对应的调用函数,确定变更函数对应的被测单元;被测单元为是待测试程序中最小的参与测试的程序模块;

[0228] 基于用例生成策略,生成被测单元的测试用例。

[0229] 根据本公开的一个或多个实施例,基于用例生成策略,生成被测单元的测试用例,包括:

[0230] 为被测单元生成测试参数,获得被测单元对应的至少一个测试参数;

[0231] 基于参数数量最小且对被测单元的代码覆盖率最大的参数选择条件,从至少一个测试参数中选择目标测试参数;

[0232] 利用目标测试参数,生成被测单元的测试用例。

[0233] 根据本公开的一个或多个实施例,为被测单元生成测试参数,获得被测单元对应的至少一个测试参数,包括:

[0234] 根据变更函数和变更函数对应调用函数之间的函数调用关系,确定被测单元对应的至少一个分支路径;

[0235] 为分支路径生成至少一个测试参数,以获得至少一个分支路径分别对应的至少一个测试参数;

[0236] 确定至少一个分支路径分别对应的至少一个测试参数为被测单元对应的至少一个测试参数。

[0237] 根据本公开的一个或多个实施例,分支路径包括变更函数中执行任一次时通过的

分支代码和分支代码在调用函数对应的调用代码;为分支路径生成至少一个测试参数,包括:

[0238] 根据变更函数的函数信息,确定分支路径在分支代码对应的第一参数;基于参数构造器,利用函数调用关系中分支代码和分支代码在调用函数对应的调用代码之间的跳转关系,确定调用代码对应的第二参数;

[0239] 确定分支代码对应的第一参数和调用代码对应的第二参数为分支路径的测试参数,返回至根据变更函数的函数信息,确定分支路径在分支代码对应的第一参数的步骤继续执行,直至为分支路径生成至少一个测试参数的参数数量满足预设的数量阈值。

[0240] 根据本公开的一个或多个实施例,基于参数数量最小且对被测单元的代码覆盖率最大的参数选择条件,从至少一个测试参数中选择目标测试参数,包括:

[0241] 对测试参数进行参数测试处理,获得至少一个测试参数分别对应的测试结果和代码覆盖信息;

[0242] 根据至少一个测试参数分别对应的测试结果,从至少一个测试参数中选择满足测试条件的候选参数;

[0243] 根据候选参数对应的代码覆盖信息,从候选参数中选择参数选择条件的目标测试参数;参数选择条件包括:参数数量最小且对被测单元的代码覆盖率最大。

[0244] 根据本公开的一个或多个实施例,对测试参数进行参数测试处理,获得至少一个测试参数分别对应的测试结果和代码覆盖信息,包括:

[0245] 获取为被测单元的代码执行覆盖参数插桩处理获得的目标测试单元;

[0246] 为目标测试单元封装程序执行接口;

[0247] 将测试参数输入执行程序接口,利用封装于程序执行接口内的目标测试单元对测试参数执行测试处理,获得测试参数对应的测试结果和代码覆盖信息;

[0248] 获取至少一个测试参数分别输入程序执行接口,以获得至少一个测试参数分别对应的测试结果和代码覆盖信息。

[0249] 根据本公开的一个或多个实施例,将测试参数输入执行程序接口,利用封装于程序执行接口内的目标测试单元对测试参数执行测试处理,获得测试参数对应的测试结果和代码覆盖信息,包括:

[0250] 将测试参数输入执行程序接口,利用封装于程序执行接口内的目标测试单元对测试参数执行测试处理;

[0251] 若确定目标测试单元对测试参数执行中断或者编译失败,则确定测试失败为测试参数对应的测试结果;

[0252] 若确定接收到目标测试单元对测试参数正常返回信息,则获取目标测试单元返回的信息为测试结果;

[0253] 根据插桩处理的覆盖参数进行数据读取,获得覆盖参数对应的覆盖数据;

[0254] 根据覆盖参数对应的覆盖数据,生成测试参数的代码覆盖信息。

[0255] 根据本公开的一个或多个实施例,测试结果包括测试成功或者测试失败;

[0256] 根据至少一个测试参数分别对应的测试结果,从至少一个测试参数中选择满足测试条件的候选参数,包括:

[0257] 根据至少一个测试参数分别对应的测试结果,从至少一个测试参数中选择测试结

果为测试成功的正常测试参数；

[0258] 根据被测单元的断言程序对正常测试参数的返回信息进行断言测试，获得正常测试参数的断言结果；

[0259] 从断言结果中确定断言正常的正常测试参数为候选参数。

[0260] 根据本公开的一个或多个实施例，根据候选参数对应的代码覆盖信息，从候选参数中选择参数数量最小且对被测单元的代码覆盖率最大的目标测试参数，包括：

[0261] 确定以参数数量最小且对被测单元的代码覆盖率最大作为求解目标的约束求解模型；

[0262] 将候选参数对应的代码覆盖信息输入约束求解模型，通过约束求解模型求解获得候选参数中满足求解目标的目标测试参数。

[0263] 根据本公开的一个或多个实施例，利用目标测试参数，生成被测单元的测试用例，包括：

[0264] 确定用例生成模板；

[0265] 将目标测试参数插入至用例生成模板对应的测试模块，并将用例生成模板中的用例参数更新为被测单元的单元信息，获得被测单元的测试用例。

[0266] 根据本公开的一个或多个实施例，将目标测试参数插入至用例生成模板对应的测试模块，并将用例生成模板中的用例参数更新为被测单元的单元信息，获得被测单元的测试用例，包括：

[0267] 将目标测试参数插入至用例生成模板对应的测试模块，并将用例生成模板中的用例参数更新为被测单元的单元信息，获得被测单元的可执行文件；

[0268] 基于被测单元的可执行文件，建立测试用例的生成消息队列；

[0269] 响应于线程或进程对生成消息队列的消费请求，从消息队列中消费可执行文件对应的执行进程，获得可执行文件执行结束时的测试用例。

[0270] 根据本公开的一个或多个实施例，待测试程序中的被测单元包括至少一个；基于用例生成策略，生成被测单元的测试用例之后，还包括：

[0271] 确定至少一个被测单元分别对应的测试用例；

[0272] 根据至少一个被测单元分别对应的文件信息，将属于同一文件信息的被测单元划分至至同一单元组，获得至少一个单元组；单元组包括至少一个测试单元；

[0273] 将单元组内的至少一个测试单元分别对应的测试用例进行用例合并，获得至少一个文件信息分别对应的文件用例；

[0274] 确定至少一个文件信息分别对应的文件用例为待测试程序的目标用例集合。

[0275] 根据本公开的一个或多个实施例，从待测试程序中解析获得相对原始程序发生代码变更的变更函数，包括：

[0276] 响应于针对待测试程序的用例生成请求，从待测试程序中解析获得相对原始程序发生代码变更的变更函数；

[0277] 基于用例生成策略，生成被测单元的测试用例之后，还包括：

[0278] 输出为被测单元生成的测试用例。

[0279] 根据本公开的一个或多个实施例，基于用例生成策略，生成被测单元的测试用例之后，还包括：

[0280] 响应于针对待测试程序的程序测试请求,利用被测单元的测试用例对待测试程序进行测试处理,获得测试结果;

[0281] 输出待测试程序的测试结果。

[0282] 根据本公开的一个或多个实施例,还包括:

[0283] 获取待测试程序对应的函式呼叫图;函式呼叫图包括待测试程序中至少一个函数之间的相互调用关系;

[0284] 从函式呼叫图中查询与变更函数存在调用关系的调用函数。

[0285] 第二方面,根据本公开的一个或多个实施例,提供一种测试用例生成装置,包括:

[0286] 程序解析单元,用于从待测试程序中解析获得相对原始程序发生代码变更的变更函数;

[0287] 调用分析单元,用于根据变更函数和变更函数对应的调用函数,确定变更函数对应的被测单元;被测单元为是待测试程序中最小的参与测试的程序模块;

[0288] 用例生成单元,用于基于用例生成策略,生成被测单元的测试用例。

[0289] 第三方面,根据本公开的一个或多个实施例,提供了一种电子设备,包括:至少一个处理器和存储器;

[0290] 存储器存储计算机执行指令;

[0291] 至少一个处理器执行存储器存储的计算机执行指令,使得至少一个处理器执行如上第一方面以及第一方面各种可能的设计的测试用例生成方法。

[0292] 第四方面,根据本公开的一个或多个实施例,提供了一种计算机可读存储介质,计算机可读存储介质中存储有计算机执行指令,当处理器执行计算机执行指令时,实现如上第一方面以及第一方面各种可能的设计的测试用例生成方法。

[0293] 第五方面,根据本公开的一个或多个实施例,提供了一种计算机程序产品,包括计算机程序,计算机程序被处理器执行时实现如上第一方面以及第一方面各种可能的设计的测试用例生成方法。

[0294] 以上描述仅为本公开的较佳实施例以及对所运用技术原理的说明。本领域技术人员应当理解,本公开中所涉及的公开范围,并不限于上述技术特征的特定组合而成的技术方案,同时也应涵盖在不脱离上述公开构思的情况下,由上述技术特征或其等同特征进行任意组合而形成的其它技术方案。例如上述特征与本公开中公开的(但不限于)具有类似功能的技术特征进行互相替换而形成的技术方案。

[0295] 此外,虽然采用特定次序描绘了各操作,但是这不应当理解为要求这些操作以所示出的特定次序或以顺序次序执行来执行。在一定环境下,多任务和并行处理可能是有利的。同样地,虽然在上面论述中包含了若干具体实现细节,但是这些不应当被解释为对本公开的范围的限制。在单独的实施例的上下文中描述的某些特征还可以组合地实现在单个实施例中。相反地,在单个实施例的上下文中描述的各种特征也可以单独地或以任何合适的子组合的方式实现在多个实施例中。

[0296] 尽管已经采用特定于结构特征和/或方法逻辑动作的语言描述了本主题,但是应当理解所附权利要求书中所限定的主题未必局限于上面描述的特定特征或动作。相反,上面所描述的特定特征和动作仅仅是实现权利要求书的示例形式。

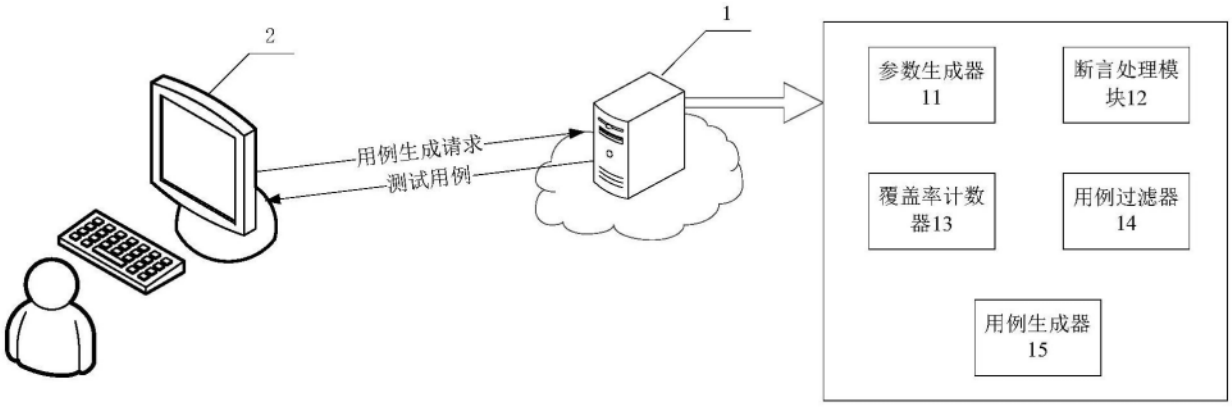

2
1
用例生成请求
测试用例
参数生成器
11
断言处理模
块12
覆盖率计数
器13
用例过滤器
14
用例生成器
15

图1

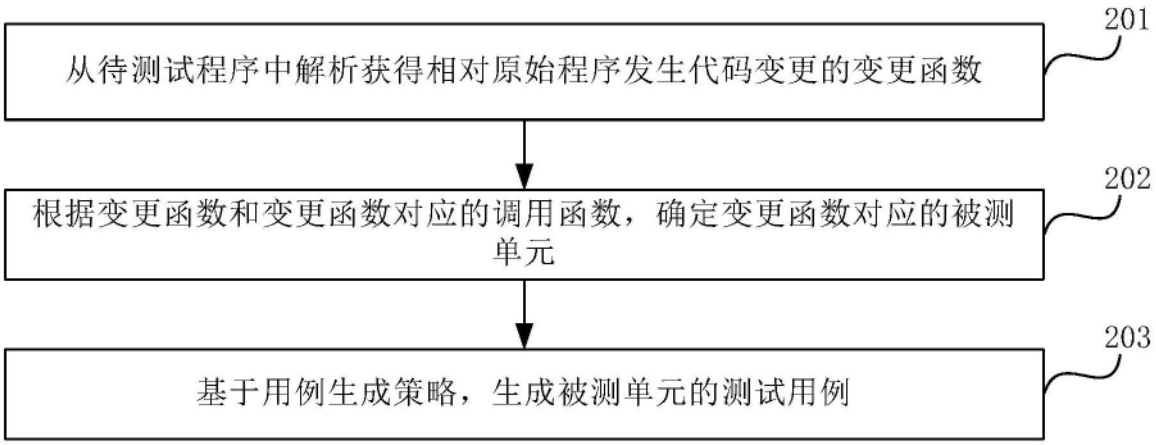

201
从待测试程序中解析获得相对原始程序发生代码变更的变更函数
202
根据变更函数和变更函数对应的调用函数，确定变更函数对应的被测单元
203
基于用例生成策略，生成被测单元的测试用例

图2

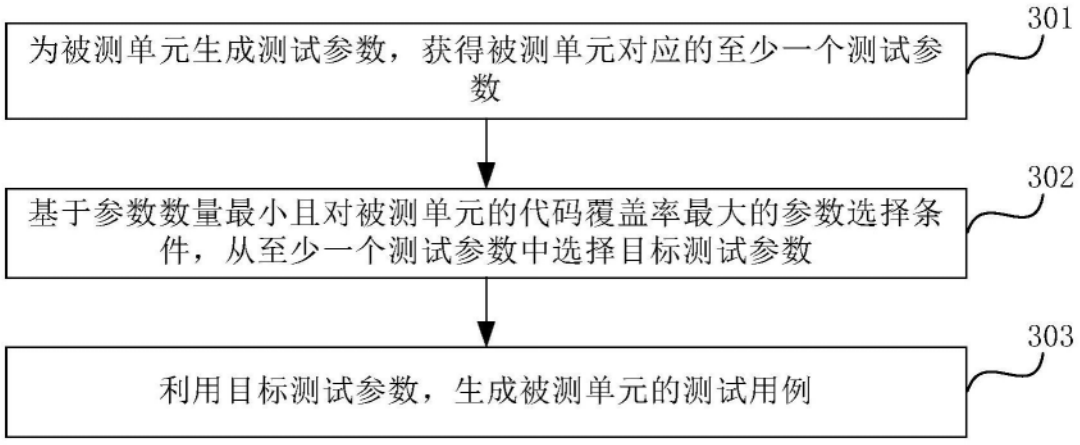

301
为被测单元生成测试参数，获得被测单元对应的至少一个测试参数
302
基于参数数量最小且对被测单元的代码覆盖率最大的参数选择条件，从至少一个测试参数中选择目标测试参数
303
利用目标测试参数，生成被测单元的测试用例

图3

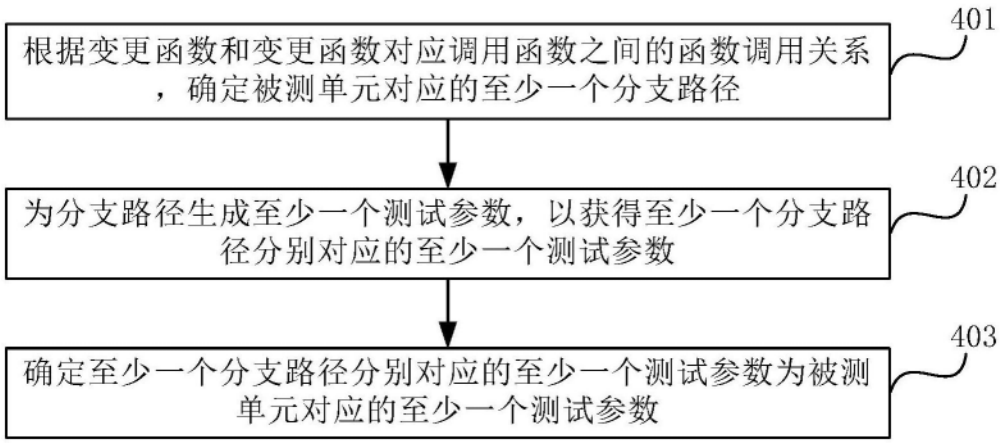

根据变更函数和变更函数对应调用函数之间的函数调用关系，确定被测单元对应的至少一个分支路径
401
为分支路径生成至少一个测试参数，以获得至少一个分支路径分别对应的至少一个测试参数
402
确定至少一个分支路径分别对应的至少一个测试参数为被测单元对应的至少一个测试参数
403

图4

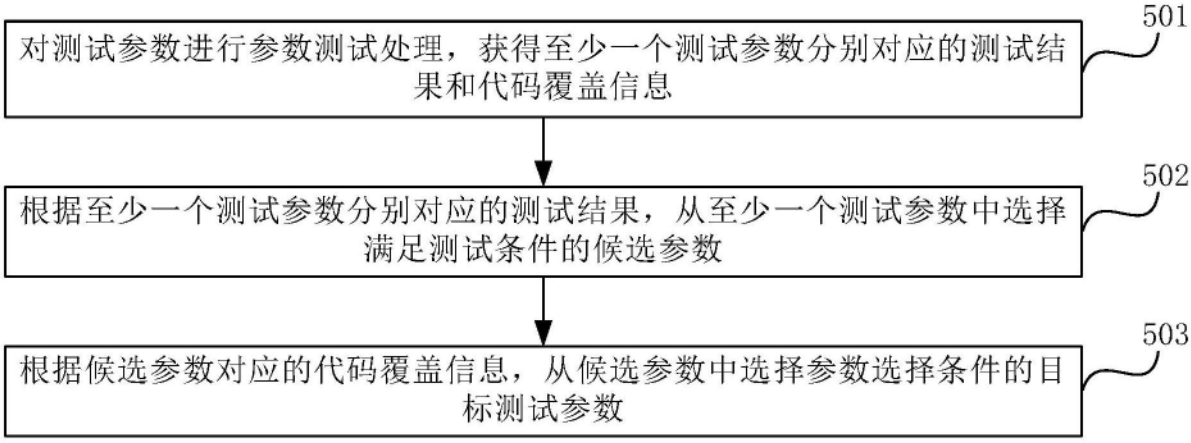

对测试参数进行参数测试处理，获得至少一个测试参数分别对应的测试结果和代码覆盖信息
501
根据至少一个测试参数分别对应的测试结果，从至少一个测试参数中选择满足测试条件的候选参数
502
根据候选参数对应的代码覆盖信息，从候选参数中选择参数选择条件的目标测试参数
503

图5

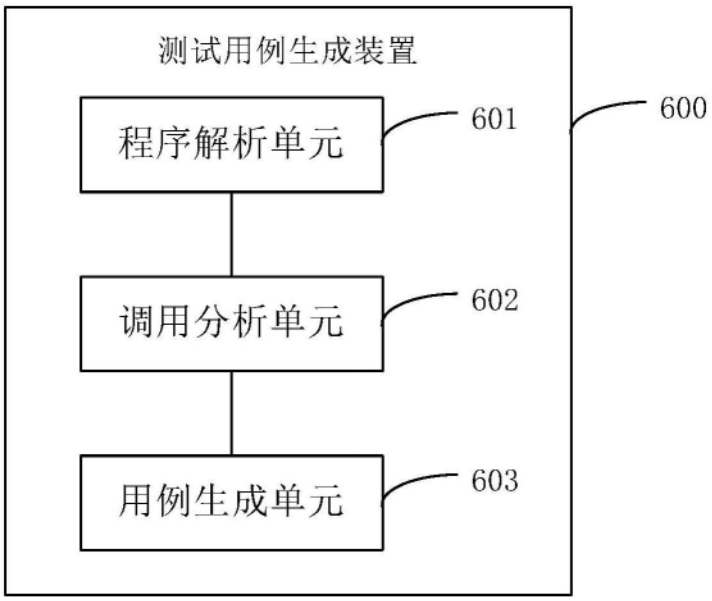

测试用例生成装置
程序解析单元
601
600
调用分析单元
602
用例生成单元
603

图6

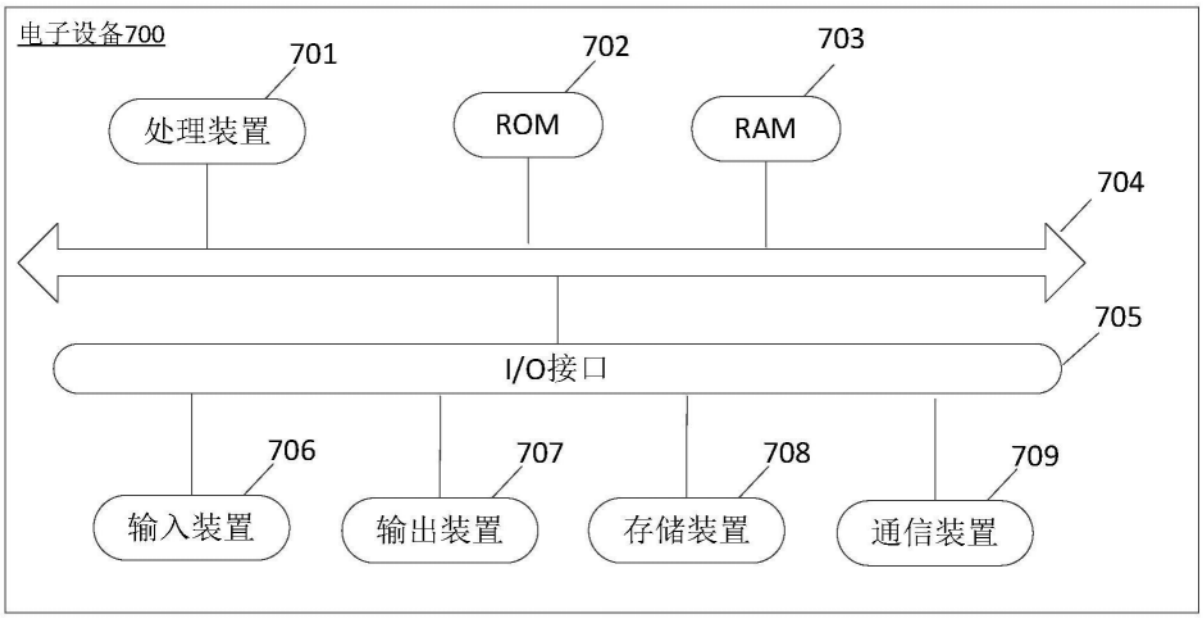

电子设备700
701
处理装置
702
ROM
703
RAM
704
705
I/O接口
706
输入装置
707
输出装置
708
存储装置
709
通信装置

图7