



(12) 发明专利申请

(10) 申请公布号 CN 103294596 A

(43) 申请公布日 2013. 09. 11

(21) 申请号 201310196434. 9

(22) 申请日 2013. 05. 23

(71) 申请人 西安电子科技大学

地址 710071 陕西省西安市太白南路 2 号西安电子科技大学

(72) 发明人 段振华 刘艳艳 田聪 张南
王小兵

(74) 专利代理机构 北京科亿知识产权代理事务所(普通合伙) 11350

代理人 汤东凤

(51) Int. Cl.

G06F 11/36(2006. 01)

G06F 11/32(2006. 01)

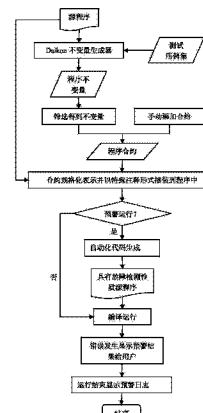
权利要求书2页 说明书5页 附图2页

(54) 发明名称

一种基于程序不变量的合约式软件故障预警方法

(57) 摘要

一种基于程序不变量的合约式软件故障预警方法，(1) 使用 Daikon 工具为需要预警的程序生成程序不变量，根据测试用例集在 Daikon 工具上运行源程序产生不变量，输出程序不变量；(2) 筛选不变量；(3) 手动生成由布尔断言组成的数据合约；(4) 将上述步骤 2 和 3 中两种方式得到的合约以规格化进行表示，并将规格化表示后的所述合约以注释的方式插桩到源程序的相应位置；(5) 将步骤 4 中插桩到相应位置的注释转换为具有故障检测性质的代码，并插桩到源程序的相应目标位置；(6) 运行经步骤 1 至 5 处理后的源程序，如果程序运行过程中违反了程序合约，则故障检测代码会自动把监测到的故障展示给用户。



1. 一种基于程序不变量的合约式软件故障预警方法,其特征在于:其包括如下步骤:

(1) 使用 Daikon 工具为需要预警的程序生成程序不变量,将需要预警的源程序和对应于源程序的测试用例集作为 Daikon 工具输入,根据测试用例集在 Daikon 工具上运行源程序产生不变量,输出程序不变量;

(2) 筛选所述不变量,所述筛选方式包括自动筛选和用户手动筛选,并将筛选后的不变量作为合约的一种来源;

(3) 手动生成由布尔断言组成的数据合约,所述数据合约是数据流的行为约束;

(4) 将上述步骤 2 和 3 中两种方式得到的合约以规格化进行表示,并将规格化表示后的所述合约以注释的方式插桩到源程序的相应位置;

(5) 将步骤 4 中插桩到相应位置的注释转换为具有故障检测性质的代码,并插桩到源程序的相应目标位置;

(6) 运行经步骤 1 至 5 处理后的源程序,如果程序运行过程中违反了程序合约,则故障检测代码会自动把监测到的故障展示给用户,表示有故障产生,并将整个运行过程中预警到的故障记录到预警日志中。

2. 如权利要求 1 所述的基于程序不变量的合约式软件故障预警方法,其特征在于:所述步骤 2 中不变量的筛选过程包括手动筛选和自动筛选:

自动筛选方式:采用 Daikon 内置的不变式过滤工具进行过滤;

手动筛选方式:将程序中所有程序点的不变量显示给用户,用户可以手动选择是否采用该程序点的不变量。

3. 如权利要求 1 所述的基于程序不变量的合约式软件故障预警方法,其特征在于:所述规格化合约的类型包括方法前置条件和后置条件型合约、类不变量型合约、循环不变量型合约以及其他类型合约。

4. 如权利要求 3 所述的基于程序不变量的合约式软件故障预警方法,其特征在于:所述方法前置条件和后置条件型合约包括方法前置条件及方法后置条件,所述方法前置条件是在方法入口处具有的性质,即当程序发生错误时,违反前置条件,且所述前置条件在方法入口处插入源程序;

所述后置条件是在方法出口处具有的性质,当程序发生错误时,违反后置条件,且所述后置条件在方法出口处插入源程序。

5. 如权利要求 3 所述的基于程序不变量的合约式软件故障预警方法,其特征在于:所述类不变量型合约为在整个类中保持不变的性质,类中每个方法调用时都要对该性质进行检查是否违反该性质,且所述类不变量型合约在类体结束处插入源程序。

6. 如权利要求 3 所述的基于程序不变量的合约式软件故障预警方法,其特征在于:所述循环不变量型合约用于找出在循环中出现的错误,且所述循环不变量型合约插入循环头及循环体之间。

7. 如权利要求 3 所述的基于程序不变量的合约式软件故障预警方法,其特征在于:所述其他类型合约为不属于上述方法前置条件和后置条件型合约、类不变量型合约、循环不变量型合约的其他合约,所述其他类型合约通过用户手动选择其插入源程序的位置。

8. 如权利要求 1 所述的基于程序不变量的合约式软件故障预警方法,其特征在于:所述步骤 5 中将合约注释均转换为具有故障检测性质的代码,并插桩到源程序的相应目标位

置,算法步骤如下:

- 1). 遍历程序中每一处注释类型;
- 2). 根据注释类型找出该合约类型以及估计的该注释类型的插桩位置;
- 3). 使用源代码插桩技术,在不破坏源代码结构的基础上,找到确定的插桩位置,在该处插入具有故障检测性质的代码。

一种基于程序不变量的合约式软件故障预警方法

技术领域

[0001] 本发明属于软件安全保障领域,具体涉及一种基于程序不变量的合约式软件故障预警方法。

背景技术

[0002] 当今随着电子计算机技术的快速进步,计算机在工业、交通、金融、医疗、通信、航空航天等领域的日益广泛应用,采用故障检测来保证程序的正确性和可靠性已经变得非常关键。

[0003] 不幸的是,随着软件规模变大,功能越来越强,故障检测也相当复杂。在大型的软件系统中,尽管经过认真的设计、开发和完备的测试,往往仍然存在许多故障。这些故障在正常运行的时候可能并没有发生错误或表现出错误的征兆,但在特定的条件下,会对系统产生破坏。如果不及时发现,随着时间的推移会发生严重的错误,甚至导致系统崩溃。

[0004] 对于软件故障的检测分为静态预测和动态预警。传统的检测方法一般是前者,该方法不要求软件实际运行,而针对其编程风格是否规范,逻辑表达式是否正确,类型匹配保证一致等检查来发现错误,但是很多其他错误,用这种静态预测的方法很难检测到,因为它们包含程序的动态行为,动态测试通常用来发现这类错误,有些错误并不能在所在测试用例环境下表现出来,因此需要故障预警来在线检测,同时故障预警作为保障软件安全的重要分支,主要用于保障软件运行时的行为符合人们的预期,越来越受到研究者的关注。

[0005] 现阶段,存在的软件故障预警技术有:通过检测读取未初始化内存错误、内存泄露和数组越界等来在线检测异常。对 C 程序增加在线检测功能,主要检测数组下标和指针越界等;对于 java 语言来说,由于 java 语言是类型安全语言以及具有垃圾回收机制,所以很多上述方法中的错误 java 程序可以自动避免,所以对于 java 语言来说现有的故障预警技术相对较少。

[0006] 目前,代表性的软件故障预警技术有:

[0007] 1. 基于合约的程序验证方法

[0008] 基于合约的程序验证是一种有效提高程序质量的技术,合约设计最早是由 meyer 针对程序设计语言 Eiffel 提出的,该技术是在程序执行的过程中动态检查是否违反程序合约,其中程序合约是具有特定形式的断言的格式,而这些断言是需要程序员去发现的,程序员在编写程序过程中发现程序遵循什么样的合约,并将其注释到 java 程序中,然后在 java 程序执行的过程中去验证是否符合这些断言规则,来发现程序异常。

[0009] 该方法的优点在于给出了一种针对 java 程序的动态故障预警方法,把程序要符合的合约提前插入到程序中,在程序运行中是否符合这些合约作为判定程序是否存在故障的准则。缺点在于该方法中的合约需要程序员手动添加,不适合对大型系统进行验证,且没有提供系统化的合约来源。

[0010] 2. 基于程序不变量的验证方法

[0011] 不变式的概念在许多领域中有着重要应用。程序理论中的不变式简单说,是指控

制流每次到达程序的某个或某些特定位置时总成立的属性。程序不变式包括循环不变式、前置不变式、后置不变式和类不变式。程序不变量常用于程序设计、程序理解、故障定位以及程序验证等领域。

[0012] 采用程序不变量进行验证时,首先基于一定的测试用例,对于程序修改前后的两个版本分别检测不变式;然后比较两个版本在相同程序点上的不变式,若所有的不变式均一致则认为对两个版本的程序验证成功。

[0013] 该方法的优点在于将不变量用于程序验证,在程序版本发生变化后,验证是相对有效的;缺点在于该方法不适用于一般的无版本变更的系统。

[0014] 3. 动态不变量发现方法

[0015] 动态不变量发现方法是通过程序实际的重复运行发现其属性的不变式性质,相对于静态不变量发现方法,动态不变量发现方法通常需要一个测试用例集,程序通过在测试用例集上反复运行来发现不变量,所以测试用例集的选取在一定程度上决定了不变量的准确性。

[0016] 随着动态不变量技术的发展,出现一些不变量发现工具,其中 Daikon 工具具有很好的发现能力,Daikon 是美国麻省理工学院计算机科学实验室的项目,利用程序执行来发现可能的不变式。它通过对源程序输入一组测试用例集,在测试用例集上运行程序,从而利用跟踪得来的结果来推导不变式。Daikon 工具已被广泛应用于研究中,如在工具 Eclat 和 Substra 中都有使用。

[0017] 该技术的优点在于已存在优秀的工具可以利用,该工具能产生比较好的不变式,可以在预警时作为程序合约使用,给合约式程序验证方法提供系统化合约来源;缺点在于生成的不变式并不完全适用于合约式程序验证,所以需要对通过工具产生的不变量进行筛选,以提高准确率。

发明内容

[0018] 本发明的目的在于克服上述已有技术的不足,并结合已有技术的优点,提供一种基于程序不变量的合约式故障预警方法,把不变量有效应用于软件故障预警,对软件进行运行时监督,对故障在线预警,保障软件安全,降低软件失效率。

[0019] 为了实现上述目的,本发明对基于合约的程序验证进行了改进,结合了动态不变量发现技术设计了软件故障预警流程,将程序不变量应用于故障预警中,最后将故障预警结果记录到预警日志中,该方法主要针对 java 程序进行预警,具体采用的技术方案如下:

[0020] 一种基于程序不变量的合约式软件故障预警方法,其包括如下步骤:

[0021] (1) 使用 Daikon 工具为需要预警的程序生成程序不变量,将需要预警的源程序和对应于源程序的测试用例集作为 Daikon 工具输入,根据测试用例集在 Daikon 工具上运行源程序产生不变量,输出程序不变量;

[0022] (2) 筛选所述不变量,所述筛选方式包括自动筛选和用户手动筛选,并将筛选后的不变量作为合约的一种来源;

[0023] (3) 手动生成由布尔断言组成的数据合约,所述数据合约是数据流的行为约束;

[0024] (4) 将上述步骤 2 和 3 中两种方式得到的合约以规范化进行表示,并将规范化表示后的所述合约以注释的方式插桩到源程序的相应位置;

[0025] (5) 将步骤 4 中插桩到相应位置的注释转换为具有故障检测性质的代码，并插桩到源程序的相应目标位置；

[0026] (6) 上述步骤是对源程序进行预处理，经过预处理的程序同时包含程序本身源代码和添加的故障检测代码，对该程序进行编译运行，如果程序运行过程中违反了程序合约，则故障检测代码会自动把监测到的故障展示给用户，表示有故障产生，达到预警功能，最后将整个运行过程中预警到的故障记录到预警日志中。

[0027] 在上述技术方案的基础上，所述步骤 2 中不变量的筛选过程包括手动筛选和自动筛选：

[0028] 自动筛选方式：采用 Daikon 内置的不变式过滤工具进行过滤；

[0029] 手动筛选方式：将程序中所有程序点的不变量显示给用户，用户可以手动选择是否采用该程序点的不变量。

[0030] 在上述技术方案的基础上，所述规格化合约的类型包括方法前置条件和后置条件型合约、类不变量型合约、循环不变量型合约以及其他类型合约。

[0031] 在上述技术方案的基础上，所述方法前置条件和后置条件型合约包括方法前置条件及方法后置条件，所述方法前置条件是在方法入口处具有的性质，即当程序发生错误时，违反前置条件，且所述前置条件在方法入口处插入源程序；

[0032] 所述后置条件是在方法出口处具有的性质，当程序发生错误时，违反后置条件，且所述后置条件在方法出口处插入源程序。

[0033] 在上述技术方案的基础上，所述类不变量型合约为在整个类中保持不变的性质，类中每个方法调用时都要对该性质进行检查是否违反该性质，且所述类不变量型合约在类体结束处插入源程序。

[0034] 在上述技术方案的基础上，所述循环不变量型合约用于找出在循环中出现的错误，且所述循环不变量型合约插入循环头及循环体之间。

[0035] 在上述技术方案的基础上，所述其他类型合约为不属于上述方法前置条件和后置条件型合约、类不变量型合约、循环不变量型合约的其他合约，所述其他类型合约通过用户手动选择其插入源程序的位置。

[0036] 在上述技术方案的基础上，所述步骤 5 中将合约注释均转换为具有故障检测性质的代码，并插桩到源程序的相应目标位置，算法步骤如下：

[0037] 1. 遍历程序中每一处注释类型；

[0038] 2. 根据注释类型找出该合约类型以及估计的该注释类型的插桩位置；

[0039] 3. 使用源代码插桩技术，在不破坏源代码结构的基础上，找到确定的插桩位置，在该处插入具有故障检测性质的代码。

[0040] 本发明的优点在于：将采用动态不变量生成技术生成的不变量以合约的方式作为在线预警方法的检测规则，提供了合约的来源，是解决合约制定困难的有效方法，设置合约的注释形式和插桩位置，提供基于源代码的插桩算法，同时还支持手动添加合约，自动化程度很高，能有效预警，使得系统在运行期间故障检测率较高。

附图说明

[0041] 图 1 是基于程序不变量的合约式故障预警流程图；

[0042] 图 2 是 Daikon 动态不变量发现过程。

具体实施方式

[0043] 下面将结合附图对本发明作进一步的描述。

[0044] 如图 1 所示,本发明为一种基于程序不变量的合约式故障预警方法,所述方法包括以下步骤:

[0045] (1) 使用 Daikon 工具为需要预警的程序生成程序不变量,将需要预警的源程序和对应于源程序的测试用例集作为 Daikon 工具输入,根据测试用例集在 Daikon 工具上运行源程序产生不变量,输出程序不变量;其中需要输入的测试用例集是由用户提供的,生成的不变量是数据合约,主要包含三种类型:入口不变量、出口不变量和类不变量。

[0046] 如图 2 所示为 Daikon 动态不变量发现过程图, Daikon 工具接受需要分析的源程序,Daikon 的 java 语言前件对待分析程序进行处理,生成一个类型声明文件和一个修改以后添加了观察点的源程序,然后编译生成一个添加了观察点的新类,通过用户设计的测试程序,对新类多次实例化,并把观察到的样本值写入到一个数据跟踪文件中。Daikon 利用数据跟踪文件和相关数据的类型声明文件,推导出相应程序点域之间满足的不变式,并把这些不变式语句输入到一个单独的文件中。

[0047] (2) 对生成不变量进行筛选,分为对不变量进行自动筛选和提供可选操作由用户手动进行筛选,将筛选的不变量作为合约的一种来源,由于步骤 1 产生的不变量并不是完全适用,为了使得到的不变量更加准确,采用两种方式进行筛选:

[0048] 1. 采用 Daikon 内置的不变式过滤工具进行过滤;

[0049] 2. 将程序中所有程序点的不变量显示给用户,用户可以手动选择是否采用该程序点的不变量;

[0050] (3) 手动生成由布尔断言组成的数据合约,所述数据合约是数据流的行为约束;

[0051] (4) 将上述步骤 2 和 3 中两种方式得到的合约以规格化进行表示,并将规格化表示后的所述合约以注释的方式插桩到源程序的相应位置,所述规格化合约的类型包括方法前置条件和后置条件型合约、类不变量型合约、循环不变量型合约以及其他类型合约。

[0052] 1. 方法前置条件和后置条件合约

[0053] 方法前置条件是在方法入口处具有的性质,即当程序发生错误时,违反前置条件,且所述前置条件在方法入口处插入源程序。用注释形式 /*@require+ 合约 */ 表示;

[0054] 方法后置条件是在方法出口处具有的性质,当程序发生错误时,可能会违反后置条件,所以后置条件在方法 return 语句或方法的结束处。用注释形式 /*@ensure+ 合约 */ 表示。

[0055] 2. 类不变量合约

[0056] 类不变量是在整个程序或类中保持不变的性质,类中每个方法调用时都要对该性质进行检查是否违反该性质,类不变量型合约在类体结束处插入源程序。在类定义中,用注释形式 /*@class+ 合约 */ 表示。

[0057] 3. 循环不变量合约

[0058] 循环不变量是为了找出在循环中出现的典型错误,例如循环不结束问题,循环不变量放置于循环头后,循环体之前。用注释形式 /*loop+ 合约 */ 表示。

[0059] 4. 其他类型合约

[0060] 其他类型合约为不属于上述方法前置条件和后置条件型合约、类不变量型合约、循环不变量型合约的其他合约，用户手动选择其插入源程序的位置。用注释形式 /*@ normal+ 合约 */ 表示。

[0061] (5) 将步骤 4 中插桩到相应位置的注释转换为具有故障检测性质的代码，并插桩到源程序的相应目标位置；算法步骤如下：

[0062] 1. 遍历程序中每一处注释类型；

[0063] 2. 根据注释类型找出该合约类型以及该注释类型的插桩位置；

[0064] 3. 使用源代码插桩技术，在不破坏源代码结构的基础上，找到确定的插桩位置，在该处插入具有故障检测性质的代码。

[0065] (6) 上述步骤是对源程序进行预处理，经过预处理的程序同时包含程序本身源代码和添加的故障检测代码，对该程序进行编译运行，如果程序运行过程中违反了程序合约，则故障检测代码会自动把监测到的故障展示给用户，表示有故障产生，达到预警功能，最后将整个运行过程中预警到的故障记录到预警日志中。

[0066] 对于本领域的技术人员来说，可根据以上描述的技术方案以及构思，做出其它各种相应的改变以及变形，而所有的这些改变以及变形都应该属于本发明权利要求的保护范围之内。

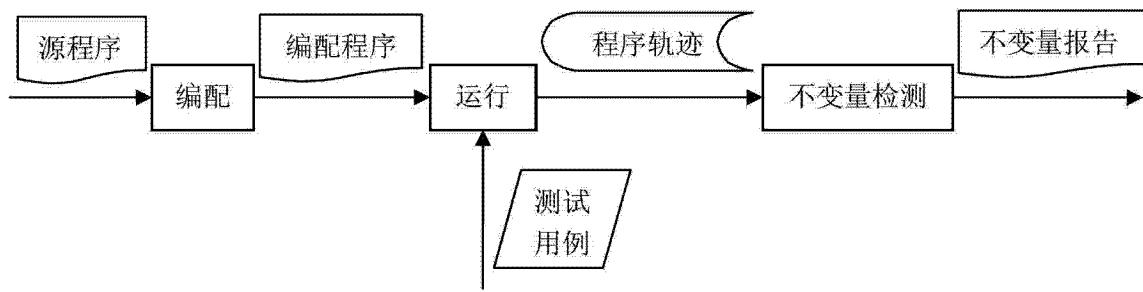


图 1

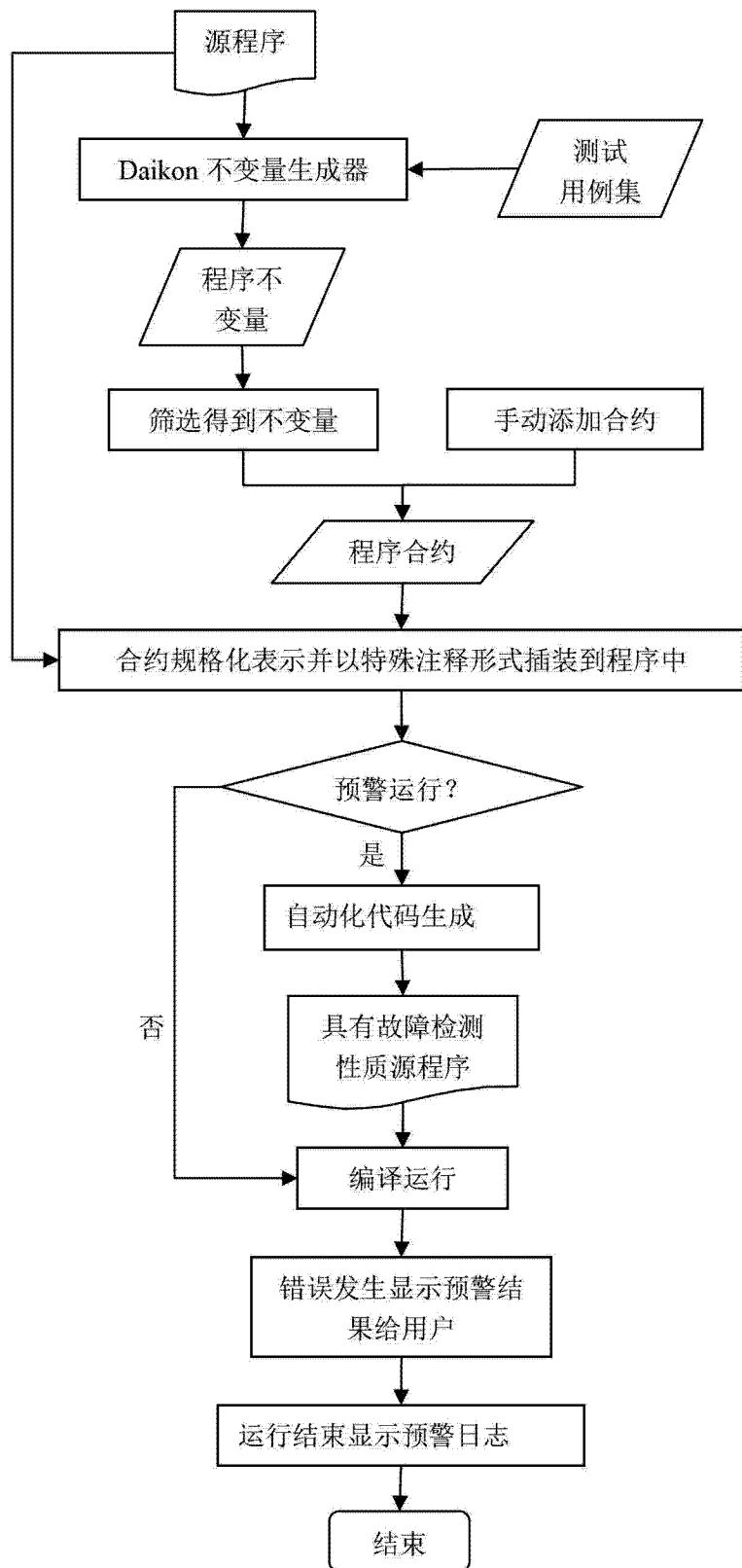


图 2