模型检测技术在软件并行缺陷检测中的应用

Application of Model Checking Technology in Software Concurrent Defect Detection

★北京广利核系统工程有限公司 孙王强

摘要：多线程并行运行的软件在提高性能的同时，其交互的组合随着程序规模增大变得更为复杂，给软件的设计与验证带来了挑战。本文从常见的并行缺陷入手，提出使用模型检测的方法对其进行检测和分析，实践表明，使用该方法可有效检测此类并行缺陷。

关键词：并发缺陷；多线程软件；模型检测

Abstract: Multithreading parallel running software not only improves the performance, but also brings challenges to software design and verification, because the combination of its interaction becomes more complex with the increase of program size. In this paper, the common concurrent defects are analyzed and verified by using model checking method. The practice shows that this method can effectively this kind of concurrent defects.

Key words: Concurrent defects; Multithreading software; Model check

1 引言

和传统的串行运行软件相比，多线程并行运行的软件在提升计算性能的同时也给软件的设计与验证带来了挑战。交互的组合随着程序规模的增大，执行时间呈指数级爆炸式增长，即使是经验丰富的开发人员，也很难开发出既正确又高效的并行软件[1]，由软件并行执行引入的、由线程调度顺序导致的缺陷称为“并行缺陷”。这些并行缺陷容易引入且难以检测。

2 软件并行缺陷特点

多线程软件通过通信实现协作，共同完成预定任务。相对于系统以单纯的串行方式工作所表现出的软件行为而言，多线程软件在设计时，不仅要考虑单个线程的逻辑处理过程，还需要考虑多个线程之间的调度顺序，但是受操作系统调度策略、线程间通信等多种因素的影响，线程调度顺序通常是不确定的。多线程软件运行时存在许多并行活动的行为轨迹，导致软件执行交互过程中，形成的状态空间庞大，并行缺陷仅在特定的行为轨迹下才能复现，因此，并行缺陷的检测及避免尤为困难。

并行执行的软件除继承了串行软件所有不确定因素，如随机数、堆、递归等的使用外，还引入了新的问题[2][3]，如饥饿、活锁、优先级翻转等，其中数据竞争、原子操作失效、顺序性失效、死锁是常见的四种缺陷类型[4]。

3 并行缺陷的主要验证方法

由于并行执行的多任务软件交互空间庞大，并发缺陷仅在罕见的条件下发生，传统的测试技术很难有效地暴露和揭示这些缺陷。针对这些问题，目前检测方法主要有三类[4]：随机延时扰动、更改线程优先级、Fuzzing技术，但是这些方法同样存在一定弊端：

随机延时扰动技术是并行执行的程序在进行共享内存访问和同步控制时，插入随机延时。该技术通过增加线程交互的概率增大出现并行缺陷的概率，但是不能完整地覆盖整个执行路径。

更改线程优先级技术通过更改线程优先级，在线程访问共享内存和同步控制时，挂起该线程，执行其他

线程。同随机延时扰动技术类似，可增大出现并行缺陷的概率，但是不能完整地遍历整个执行路径。

其他的还有Fuzzing技术、基于线程交织冗余分析的并行bug检测技术等，这些技术均基于动态监测的原理监测并行缺陷，理论上均不能实现程序行为轨迹的全覆盖。而基于数学建模的模型检测技术在理论上可实现状态的全遍历，可有效检测并行缺陷。

3.1 模型检测

模型检测是判断硬件和软件设计的理论模型是否满足规范的方法[5]。针对状态有限系统，通过状态迁移系统表示系统行为，用模态/时序逻辑公式描述系统属性，判断系统行为（状态迁移系统）是否满足系统属性（逻辑公式），当不满足时，会给出反例。比较常见的模型检测方法有Petri网和进程代数。

Petri网缺乏对事件之间从属和时序关系描述的能力，并且Petri网较为复杂，缺少专门的工具支持，在工程实践中应用相对较少。因此选用进程代数的方法。进程代数存在三种经典的进程代数方法：CSP、CCS、Pi演算，考虑到CSP由专门的工具SPIN支持，且满足多任务并发软件的分析和验证，因此，本方法选用SPIN作为形式化分析工具，其工作原理如图1所示。

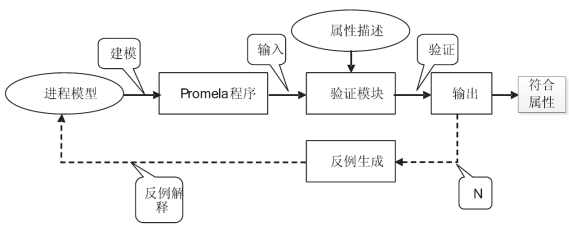

图1 模型检测的基本原理

通过SPIN工具执行形式化验证时，将每个线程视为SPIN中的一个进程，通过Promela语言描述模型行为，利用断言的形式描述协议属性，工具自动将模型行为转换为状态机，并判断模型是否满足协议属性。如不满足，则给出反例。

3.2 模型检测建模

建模的过程是通过Promela语言描述系统行为、通过时态逻辑（LTL）和断言（assertion）来声明系统属性的过程。具体分为两步：

（1）标记出相互通信的线程，按照各个线程的执行顺序建立相关进程。

组成一个协议系统的进程可以符号化为一系列状态，协议系统中的事件则可用有限状态机的输入表示；依据进程和事件之间的关系，可以得到状态和输入之间的转移关系。

（2）通过断言描述系统属性，如无死锁、无数据竞争等。

3.3 多线程软件并行缺陷检测

以验证某基于某操作系统上的多线程软件“无死锁”属性为例，说明模型检测在并行缺陷验证上的应用。

（1）描述系统行为

· 标记出通过相互通信的线程共10个，对每个线程，分别对应在SPIN中建立相应的进程P1~P10；

· 对每个线程，抽象其功能，并分别在进程P1~P10中通过Promela描述。

（2）描述系统属性

通过断言或在SPIN工具中选中safety选项。

运行后结果如图2所示。

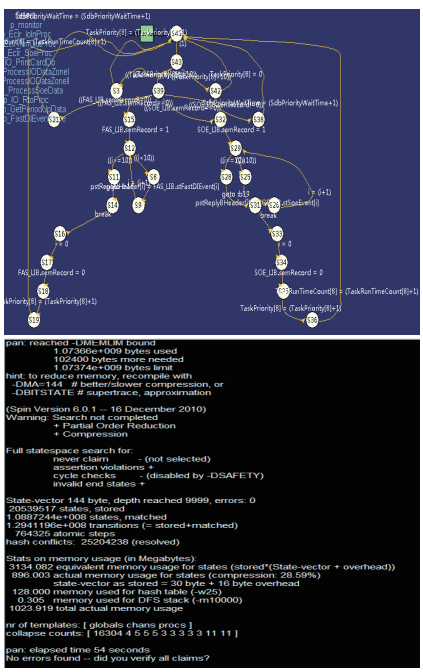

图2 多线程软件模型检测结果

4 展望及不足

本文仅对单个CPU中单核上多线程软件的并行缺陷进行分析与验证。事实上，随着现代处理器技术的发展，处理器引入的流水线、多级缓存、指令预取以及多核等技术同样存在软件并行执行的现象，会导致并行缺陷。

在指令级同样会引入不确定性。为了提高处理器的运算能力，现代处理器内部普遍使用了复杂的优化技术，如流水线、多级缓存、指令预取以及多核等技术。这些技术的使用导致指令的执行过程变得十分复杂，需要后续进一步研究。AP

作者简介：

孙王强（1979-），男，山东潍坊人，工程师，硕士，现就职于北京广利核系统工程有限公司，主要从事核安全级数字化仪控系统研究工作。

参考文献：

[1] S. Lu, S. Park, E. Seo, and Y. Zhou, Learning from Mistakes: A Comprehensive Study on Real World Concurrency Bug Characteristics[J]. ACM SIGARCH Computer Architecture News, 2008, (36) : 329 - 339.

[2] 陈沉. 面向多线程程序的确定性并行关键技术研究[D]. 湖南：国防科学技术大学, 2015.

[3] 吴振东. 并行程序中bug检测技术研究[D]. 湖南：国防科学技术大学, 2015.

[4] 禹振. 并发缺陷的检测与规避研究[D]. 黑龙江：哈尔滨工业大学, 2017.

[5] 古天龙. 软件开发的形式化方法[M]. 北京：高等教育出版社, 2005.