静态程序分析技术

MF1933108-许端琛

# 研究背景

程序编程语言方面的研究已有60年的研究历史，同时也是非常活跃的一个方向。程序编程语言的研究大致可以分为3个部分，包括理论部分，环境部分以及应用部分。其中理论部分包含语言的设计，类型系统，形式语义等。环境部分包含编译器，运行时系统等。而应用部分包含程序分析，程序验证，程序合成。其中在PL的框架下，程序分析多指静态程序分析，也即该课程报告所要讨论的主要内容。

在过去十多年，静态代码分析取得了长足的进步，并且出现了许多新颖的应用：最初主要用于运行前程序的优化，现在它也已成为理解程序，程序设计，查找软件质量缺陷，特别是安全漏洞的通用工具。 这项成功归功于这数十年来的静态分析研究，这些研究催生了新颖的算法，数据结构和设计原理，使静态分析比以往任何时候都更加精确。

而这段时间里，编程语言的核心部分改变的非常少，然而由编程语言构建的程序变的越来越大，而且越来越复杂。因此，尽管从绝对意义上说，静态分析研究的进展是显着的，但人们必须意识到，随着软件应用程序的规模和复杂性不断增加，静态分析技术的研究应该依然得到持续的突破和进步。这也是静态程序分析技术仍有很大研究价值的原因之一。

# 问题定义

静态程序分析推理计算机程序的行为而不实际运行它们。这不仅对优化编译器以产生高效代码很有用，而且对于自动错误检测和其他可以帮助程序员的工具也很有用。静态程序分析器是引起其他程序行为的程序。

众所周知，测试，即具体地运行程序并检查输出，这种方法可能会发现错误，但通常不能表明没有错误。相反，静态程序分析可以（使用正确的近似值）检查程序的所有可能执行情况，并提供有关程序属性的保证。

开发此类分析的主要挑战之一是如何确保高精度和高效率在实际中有用。例如，如果它报告许多误报，或者如果它太慢而无法适应现实世界的软件开发过程，则没有人会使用这样设计的分析。

目前程序静态分析可用于例如一下方面：

1. 程序优化分析。优化的编译器（包括解释器中的即时编译器）需要了解正在编译的程序的许多不同属性，以便生成有效的代码。
2. 程序正确性和安全性分析。设计用于检测错误（或验证是否没有错误）的最成功的分析工具都针对以特定编程语言编写的程序的通用正确性属性。在不安全的语言（如C）中，此类错误有时会导致严重的安全漏洞。在更安全的语言（如Java）中，此类错误通常不会那么严重，但它们仍可能导致程序崩溃。
3. 程序开发分析。现代IDE执行各种程序分析以支持调试，重构和程序理解。

# 方法和技术

## 中间表达

## 数据流分析

## 控制流分析

## 程序过程内和过程间分析

## 抽象表达

# 现有工具

# 研究趋势