迎新报告

刘力铭

目录

1 研究内容

我的研究内容主要是 C 和 C++ 代码的静态分析，其主题为利用静态分析发现程序中的缺陷，比如空指针接引用、内存泄露等。研究中常用到的技术包括静态符号执行、数据流分析等。我日常与代码的抽象语法树以及线性的中间表示打交道，以下是我读研以来进行的一系列研究工作。

1.1 C 规则检查器

我负责维护一个 C 代码的规则检查器——Crulet。它是前辈的毕设项目，毕业后由我维护。它扫描 C 代码，检查它是否符合 GJB 5369-2005 中的代码书写规则，并对代码中不符合规则的片段进行警告，图 1 展示了一段简单的代码以及 Crulet 对它的报告。

GJB_8_1_2.c

```
void static_p(int a){
    if(a == 1)
        a++;
}
```

```
GJB_8_1_2.c:1:6: warning: [GJB-9.1.1]
 函数必须有返回语句
void static_p(int a){
     ^
GJB_8_1_2.c:1:19: remark: [GJB-1.2.1]
 推荐在同一文件中对基本类型进行typedef
void static_p(int a){
                  ^
GJB_8_1_2.c:3:9: warning: [GJB-2.1.3]
 then/else中的语句必须用大括号括起来
        a++;
        ^
2 warnings generated.
```

图 1: 违反 GJB 5369-2005 的 C 代码

我对 Crulet 进行了大规模重构，添加了崩溃恢复和多文件并行化等实用功能，并实现了 GJB 5369-2005 中推荐类的规则相应的检查。

目前该工具已被注册了软件所的软件著作权，成为软件所版权软件的

一员。

1.2 C 符号执行分析器

我参与维护了组内重要工具 Canalyze。Canalyze 最初是由前辈和苹果合作开发，并集成到 Clang 中，之后前辈自己重构的一个独立工具。该工具利用符号执行技术检查程序中的各种缺陷，目前长于内存相关错误的检查，包括但不限于空指针解引用、内存泄漏。图 2 展示了 Canalyze 如何报告空指针解引用错误。

test.c

```
int main()
{
    int *p = 0;
    return *p;
}
```

```
1. int main()
1. Start Analysis.
2. {
3.     int *p = 0;
4.     return *p;
2. The pointer is NULL.    *p
5. }
6.
```

图 2: 一段空指针解引用代码

我参与修复了 Canalyze 的许多 BUG，使得 Canalyze 分析结果更准确，并针对购买 Canalyze 的客户的需求对 Canalyze 进行定制。同时，我还发起了 Canalyze 的迁移工作，旨在使 Canalyze 能分析 C++ 代码。

1.3 关键字指导的灰盒模糊测试工具

前辈提出了关键字指导的灰盒模糊测试方法，GTFuzz，我参与了提取关键字的算法设计，并实现了关键字候选的提取算法。关于此工作的详细描述可见将要发表于 PRDC 2020 的相应论文。

这个课题的目的是自动化地复现静态分析生成的缺陷报告，从而降低人工确认缺陷报告的成本。目前，GTFuzz 在一些输入为语义化字符串的程序上取得了较好的成果。

1.4 异常处理可达性分析

这个课题是正在进行的研究工作，旨在针对 C++ 语言分析带有异常处理相关基本块的控制信息，其中最重要的信息就是异常处理相关基本块的

可达性。由于课题正在进行中，暂时还没有什么成果。

2 自我感悟

科研不是件容易的事，这表现在好几个方面。首先，不存在任何指南，所有的事情要自己探索。单单这一点，就可以体现出科研的难度，它可能是我目前需克服的最困难的关卡。科研还经不起失败，它不是能让你不限时的做下去的，一旦失败，许多关键考核的时间点都会错过。科研还需要经受别人的置疑。导师会置疑你，可能是从你选题开始，也可能是你接受了导师的题目，但在做研究的过程中受到了置疑。评审会置疑你，这样就可能无法通过考核，或者论文无法录用。甚至你自己会置疑自己，当前的工作还能不能进行下去。最后，要知道我还没有完整地经历过一次科研，所以体会到的不容易可能不限于这些方面。

3 勉励告诫

为了大家能经得起失败，扛得住置疑，我觉得有几点建议大家可以参考。首先，尽早开始了解导师的研究方向，并开始寻找课题。开始得早，就有失败的资本。其次，研究性的工作和工程性的工作是两条路。研究性的工作讲究独创性，它不要求一个完整的工作。而工程性的工作不要求独创性，它需要的是一个实用的工具。如果要两者兼顾的话，往往得不尝失。最后，别人的置疑往往是成功的养分。我们要能分辨哪些置疑是对改进自己工作有用的，哪些是没用的，然后从有用的置疑中学习，并从无用的置疑中获得把自己工作做下去的动力。