《MUVI》——李菁菁

MUVI 是一种发现多变量访问相关性和检测多变量访问产生并发 bug 的一种方法。

MUVI 通过代码分析检测两种多变量访问错误，一种是多变量不一致更新（inconsistent updates），另一种是多变量并发错误（multivariable concurrency bugs）。

多变量不一致更新(inconsistent updates)是指如果多个变量之间存在访问关联性，更新其中一个变量后，却没有更新与它相关的变量所导致的多变量之间的不一致性问题。

多变量并发错误是指在多线程中，多个线程共享一些变量，其中至少一个线程对某个共享变量进行了写操作，造成的关联变量不同步问题

MUVI 通过分析程序的源代码来找到多个变量之间的相关性关系。相关性语义的判断是将一个函数作为为分析单元，并将变量出现在源代码中的位置距离作为变量之间的距离。如果两次访问（读或写）出现在一个函数内，并且它们之间的的距离小于阈值 MaxDistance，则将这两次访问看做是同时访问的

MUVI 中关联性访问的规定如下，在 A1(x)=>A2(y)中，当且仅当 A(x)和 A2(y)同时出现的次数最少为 MinSupport 次,而且每次 A1（x）出现，A2(y)以最少 Minconfidence 的概率出现，那么 x 和 y 具有访问关联性。

MUVI 的关联性推测包括三个步骤：

（1）收集所有访问信息，通过解析源代码来收集每个函数的变量访问信息，包括变量被访问的集合，被访问类型以及在代码中出现的位置，这些信息被存放在 Acc_Set 数据库中。MUVI 只考虑全局变量和结构体，通过访问类型（读或写）

来对不同关联类型进行分类，通过在源代码中出现的位置（文件名和在代码中出现的行数）来判断两次访问的同时性，访问来自函数本身还是它调用的函数为之后的相关性关系修剪做支持。

（2）分析访问模式，通过频繁集挖掘技术 FPclose 处理 Acc_Set 数据库中的数据，找到频繁同时出现的变量集，将这些结果作为相关性变量候选集。在频繁集挖据技术中，将一组集合作为输入，找到出现频率大于 MInSupport 阈值的子集。

（3）相关性产生、修剪和排名。利用 Acc_Set 数据中的详细信息，产生和修剪相关性集合，并对不同类型的相关性进行排名。

有以下四种情况关联关系会被去除，1）如果一种关系（C）在函数（supporter）中出现的次数不少于 MInSupport，定义为 support（C），但是满足这种关系的函数小于 support 阈值；2）如果 support(C)/support(A1(x))<MInConfidence 阈值；3）直接 supporter 的数量小于 MinDirectSupport 阈值 ;4）一些常见的变量 stdout 和 stderr 等产生的关联性。

对相关性进行排序，当 support 值很大的时候，根据 confidence 值来进行排序；若 support 值不大，则根据 suppor 进行排序。

不一致更新错误，根据关联性分析的结果集，对于任何 write(x)=>Acc(y)关系，如果在更新 x 之前没有访问 y，那么这些函数就被视为不一致更新错误候选集。MUVI 通过检查两级的函数调用和被调用关系，来去除一些假的错误候选集。之后根据一些限制对剩余的错误集进行排序。

多变量并发错误，对现有的两种经典的单变量竞争发现算法，lock-set 算法和 happens-before 算法进行扩展，来发现多变量之间竞争关系。MUVI 同样也可

以扩展其他的现有的并发错误发现工具来完成发现多变量之间的并发错误。

但是 MUVI 在函数调用关联性分析时，只分析了两级函数的调用关系，没有根据调用关系划分权重，同时它无法识别出那些 support 值很小的变量访问相关关系和条件型关联关系。而且 MUVI 针对不同的程序需要用户自己定义相关的参数，这对用户来说，不得不分析不同阈值的结果进行下一步的判断。