缺陷修复技术

熊英飞

北京大学软件工程研究所

报告人介绍 – 熊英飞

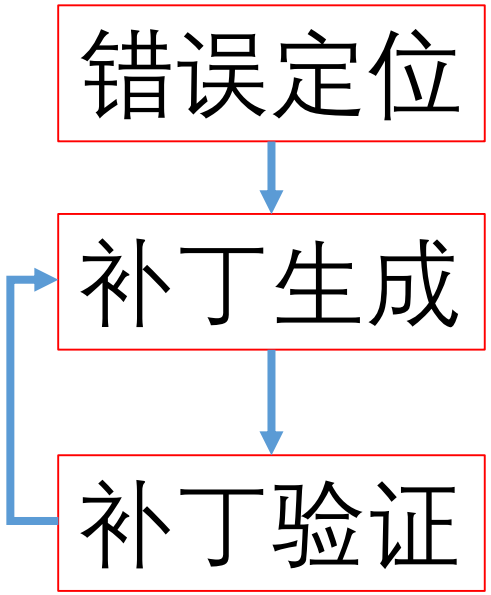
- 2000~2004，电子科技大学本科
- 2004~2006，北京大学研究生
  - 导师：梅宏、杨芙清
- 2006~2009，日本东京大学博士
  - 导师：胡振江、武市正人
- 2009~2011，加拿大滑铁卢大学博士后
  - 导师：Krzysztof Czarnecki
- 2012~，北京大学“百人计划”研究员（Tenure-Track）
- 研究方向：软件分析、编程语言设计

缘起

- 人和Bug的斗争从来没有停止过
- 缺陷检测：到底有没有Bug
  - 从上世纪60年代开始
  - 代表技术：软件测试、软件验证
- 缺陷定位：Bug在哪里
  - 从上世纪90年代开始
  - 代表技术：统计性调试
- 缺陷修复：自动消除Bug
  - 约从2000年之后开始
  - 代表技术：生成-验证缺陷修复技术

“生成-验证”缺陷修复

输入：一个程序和一组测试，至少有一个测试没有通过
输出：一个补丁，可以使程序通过所有测试



代表性工作

- GenProg
  - [Westley Weimer: ICSE’09, GECCO’09, CACM’10, ICSE’12]
  - 错误定位：采用统计性调试
  - 补丁生成：
    - 基本操作：复制其他语句/删除语句
    - 采用遗传算法从基本操作合成补丁
  - 补丁验证：运行程序中的测试验证补丁
  - 实证研究：55/105，8$/bug
- 引发一系列相关工作
  - AutoFiix, Nopol, RSRepair, MintHint, AutoRepair, SemFix, DirectFix, SPR…
- 程序员的前景一片光明，“躺着也能把钱挣了”的时代眼看就要到来

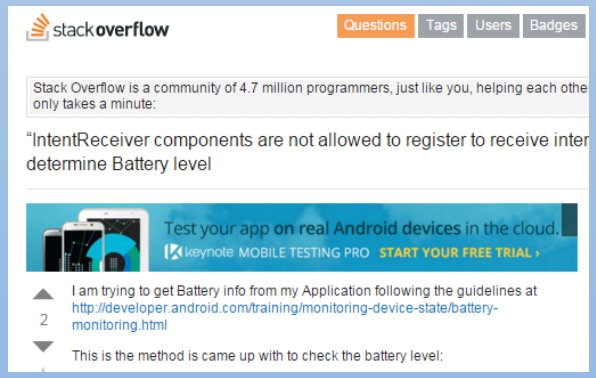
转折

- [Qi-ISSTA'15]
  - GenProg被认为修复的55个缺陷中，只有2个是正确的
  - 根本原因：通过测试并不代表是正确的修复
- [Le Goues-FSE'15]
  - 详细实验了GenProg，AE等多个主流修复方法，采用了更大的数据集，更多的测试集
  - 结果基本一致
- 其他后续工作
  - Prophet, Angelix
  - 补丁的正确率最好也不到40%

我们的工作

高正确率的缺陷修复

从QA网站学习 [ASE15]



精准条件修复 [ICSE17]

[ASE15] Qing Gao, Hansheng Zhang, Jie Wang, Yingfei Xiong, Lu Zhang, Hong Mei. Fixing Recurring Crash Bugs via Analyzing Q&A Sites. ASE'15

[ICSE17] Yingfei Xiong, Jie Wang, Runfa Yan, Jiachen Zhang, Shi Han, Gang Huang, Lu Zhang. Precise Condition Synthesis for Program Repair. ICSE'17

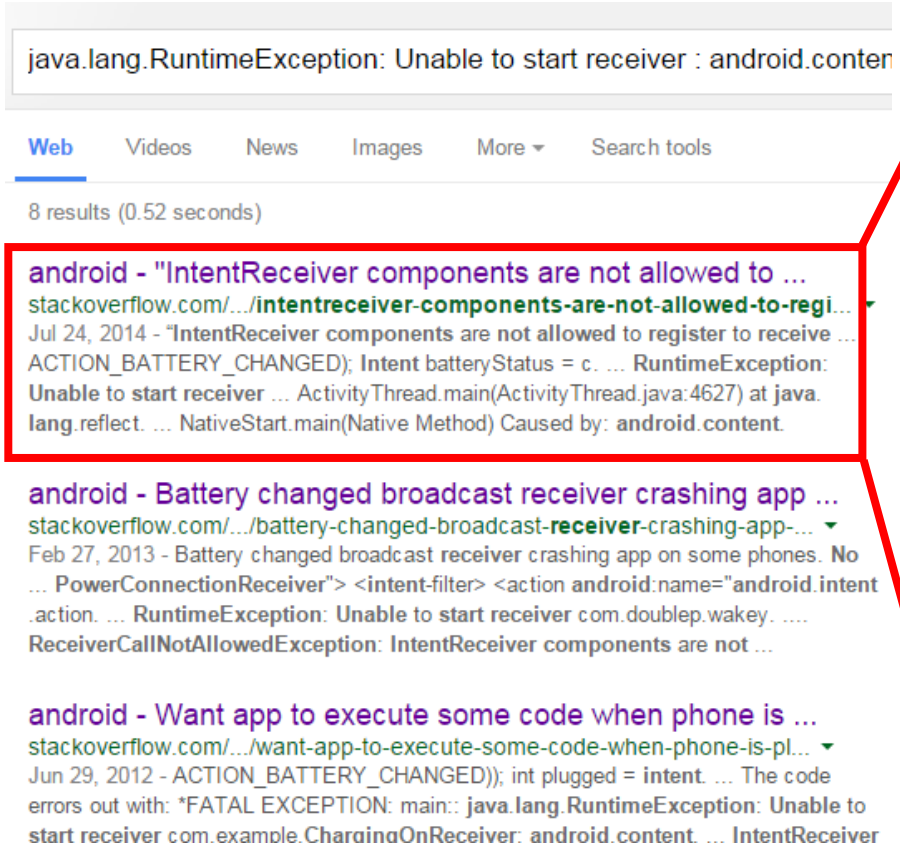
从QA网站学习

- 开发人员遇到未知错误的时候会怎么办？

```
29    public void onReceive (final Context context, final Intent intent) {
30       final int action = intent.getExtras().getInt(KEY_ACTION, -1);
31       final float bl = BatteryHelper.level(context);
32       LOG.i("AlarmReceiver invoked: action=%s bl=%s.", action, bl);
33       switch (action) {
...          ...
51       }
52    }
```

java.lang.RuntimeException: Unable to start receiver com.vaguehope.onosendai.update.AlarmReceiver:

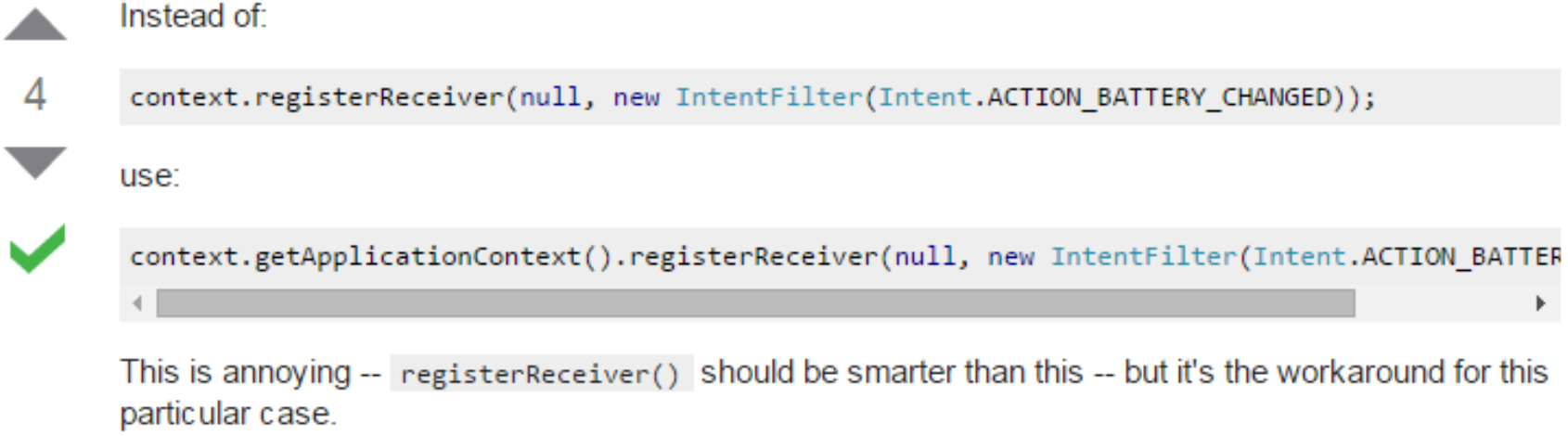
从QA网站学习


java.lang.RuntimeException: Unable to start receiver : android.conten
Web
Videos
News
Images
More
Search tools
8 results (0.52 seconds)
android - "IntentReceiver components are not allowed to ...
stackoverflow.com/.../intentreceiver-components-are-not-allowed-to-regi...
Jul 24, 2014 - "IntentReceiver components are not allowed to register to receive ... ACTION_BATTERY_CHANGED); Intent batteryStatus = c. ... RuntimeException: Unable to start receiver ... ActivityThread.main(ActivityThread.java:4627) at java.lang.reflect. ... NativeStart.main(Native Method) Caused by: android.content.
android - Battery changed broadcast receiver crashing app ...
stackoverflow.com/.../battery-changed-broadcast-receiver-crashing-app-...
Feb 27, 2013 - Battery changed broadcast receiver crashing app on some phones. No ... PowerConnectionReceiver"> <intent-filter> <action android:name="android.intent.action. ... RuntimeException: Unable to start receiver com.doublep.wakey. .... ReceiverCallNotAllowedException: IntentReceiver components are not ...
android - Want app to execute some code when phone is ...
stackoverflow.com/.../want-app-to-execute-some-code-when-phone-is-pl...
Jun 29, 2012 - ACTION_BATTERY_CHANGED)); int plugged = intent. ... The code errors out with: *FATAL EXCEPTION: main:: java.lang.RuntimeException: Unable to start receiver com.example.ChargingOnReceiver: android.content. ... IntentReceiver


stackoverflow
Questions
Tags
Users
Badges
Stack Overflow is a community of 4.7 million programmers, just like you, helping each othe
only takes a minute:
"IntentReceiver components are not allowed to register to receive inter
determine Battery level
Test your app on real Android devices in the cloud.
keynote MOBILE TESTING PRO START YOUR FREE TRIAL ›
2
2
I am trying to get Battery info from my Application following the guidelines at http://developer.android.com/training/monitoring-device-state/battery-monitoring.html
This is the method is came up with to check the battery level:
public void sendBatteryInfoMessage(){
IntentFilter iFilter = new IntentFilter(Intent.ACTION_BATTERY_
Intent batteryStatus = c.registerReceiver(null, iFilter);

从QA网站学习的困难

- 自然语言理解是很困难的



Instead of:

```
context.registerReceiver(null, new IntentFilter(Intent.ACTION_BATTERY_CHANGED));
```

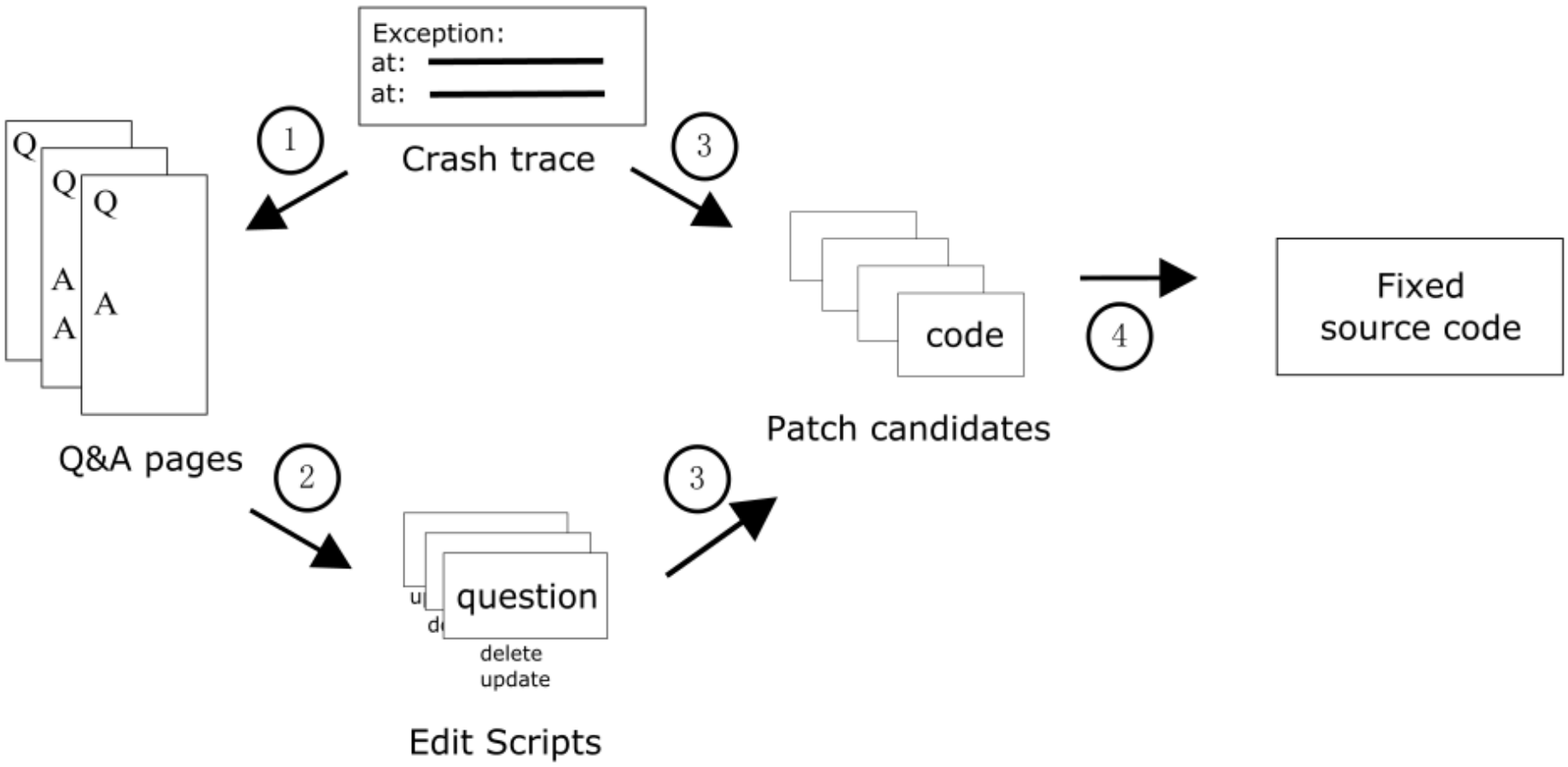
use:

```
context.getApplicationContext().registerReceiver(null, new IntentFilter(Intent.ACTION_BATTER
```

This is annoying -- `registerReceiver()` should be smarter than this -- but it's the workaround for this particular case.

- 观察：程序员常常只用编程语言语言交流的
- 解决方案：直接比较代码片段

方法概览


Exception:
at:
at:
Crash trace
1
3
Q
Q
Q
A
A
A
Q&A pages
code
Patch candidates
4
Fixed
source code
2
3
question
delete
update
Edit Scripts

实验效果

- 24个Android崩溃缺陷
  - 预先人工验证过在StackOverflow上能找到答案

- 正确修复：8
- 错误修复：2
- 正确率：80%
- 召回率：33%

精确条件修复

条件错误是很常见的

```
   lcm = Math.abs(a+b);
+ if (lcm == Integer.MIN_Value)
+   throw new ArithmeticException();
```

缺少边界检查

```
-  if (hours <= 24)
+ if (hours < 24)
      withinOneDay=true;
```

条件过强

```
-  if (a > 0)
+ if (a >= 0)
      nat++;
```

条件过弱

ACS修复系统

- ACS = Accurate Condition Synthesis
- 两组修复模板

条件修改

- 首先定位到有问题的条件，然后试图修改条件
  - 扩展：if ($D) => if ($D || $C)
  - 收缩：if ($D) => if ($D && $C)

返回预期值

- 在出错语句前插入如下语句
  - if ($C) throw $E;
  - if ($C) return $O;

挑战和解决方案

```
  int lcm=Math.abs(
     mulAndCheck(a/gdc(a,b),b));
+if (lcm == Integer.MIN_VALUE) {
+   throw new ArithmeticException();
+}
  return lcm;
```

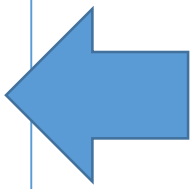
测试 1:
Input: a = 1, b = 50
Oracle: lcm = 50

测试 2:
Input: a = Integer.MIN_VALUE, b = 1
Oracle: Expected(ArithmeticException)

正确条件:
lcm == Integer.MIN_VALUE

可以通过测试的条件:

- a > 1
- b == 1
- lcm != 50
- …



排序

排序方法1:
按数据依赖对变量排序

- 变量使用局部性：最近被赋值的变量更有可能被使用。

- 根据数据依赖对变量排序
  - lcm = Math.abs(mulAndCheck(a/gdc(a, b), b))
  - lcm > a, lcm > b

排序方法2:
根据Java文档过滤变量

```
/** ...
 * @throws IllegalArgumentException if initial is not between
 * min and max (even if it <em>is</em> a root)
**/
```

抛出IllegalArgumentException时，只考虑将“initial”变量用在条件里

排序方法3:
根据现有代码对操作排序

- 在变量上使用的操作跟该条件的上下文紧密相关

变量类型

```
Vector v = …;
if (v == null) return 0;
```

变量名字

```
int hours = …;
if (hours < 24)
    withinOneDay=true;
```

方法名字

```
int factorial() {
  …
  if (n < 21) {
    …
```

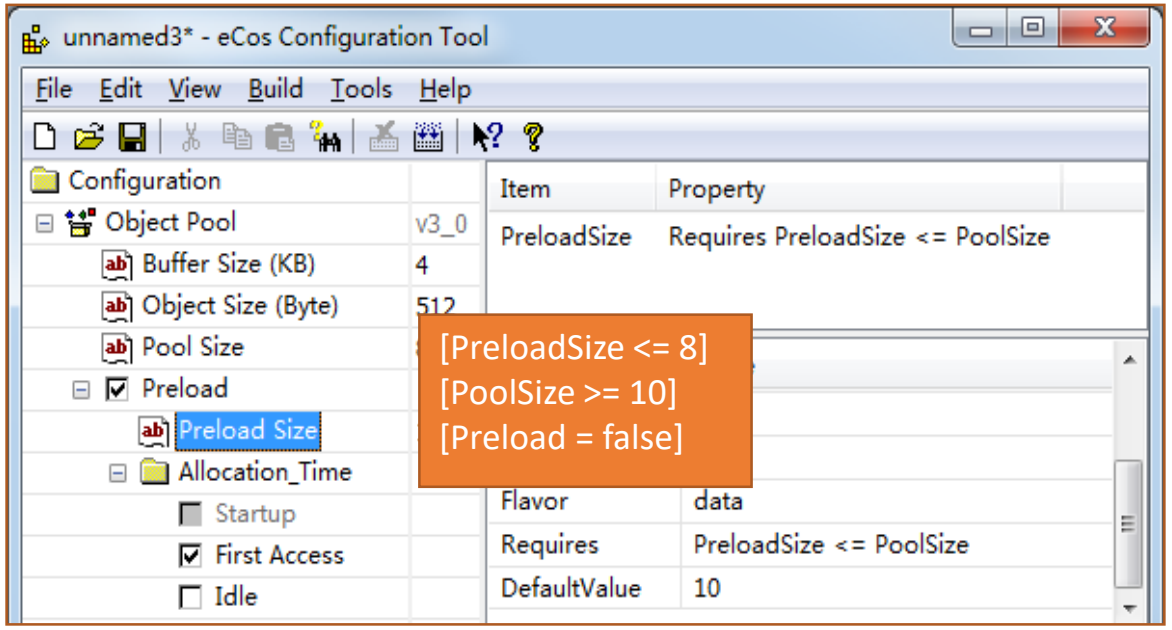
- 根据已有的代码库统计条件概率

Defects4J上的验证

Approach	Correct	Incorrect	Precision	Recall
ACS	18	5	78.3%	8.0%
jGenProg	5	22	18.5%	2.2%
Nopol	5	30	14.3%	2.2%
xPAR	3	–[4]	–[4]	1.3%[2]
HistoricalFix[1]	10(16)[3]	–[4]	–[4]	4.5%(7.1%)[2,3]

其他成果: 软件配置交互式修复

- Linux内核包含6000余条配置项，1000余条约束
- 嵌入式操作系统eCos包含1000余配置项，600余条约束
- 调研表明：用户往往不知如何修复配置中的错误。

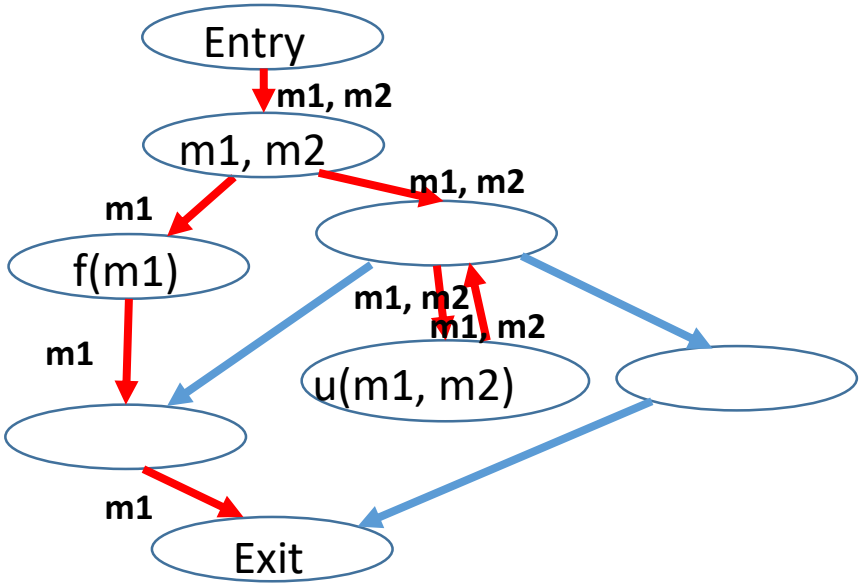


全自动生成修复列表，首次同时保证修复的正确性、最优性和完整性。

成果发表于IEEE Transactions on Software Engineering，并被选为网站首页论文

其他成果:内存泄露自动修复

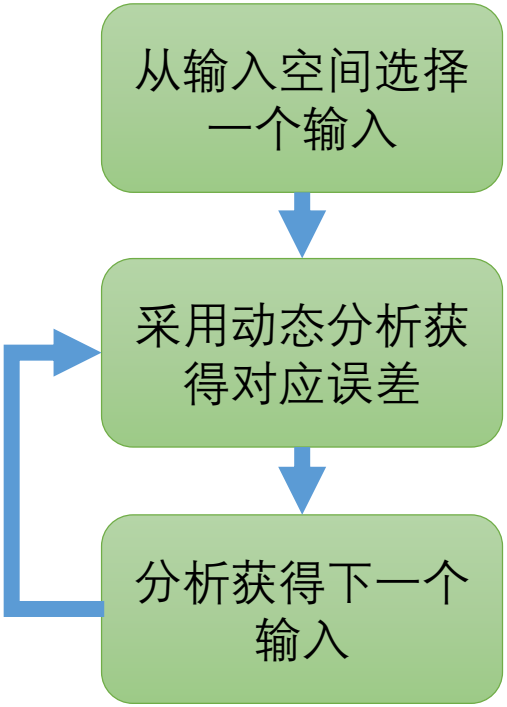
- 大量安全攸关软件采用C语言开发
- 内存泄露是C语言的一大难题
- 虽然有大量内存泄露检测技术，内存泄露的修复仍然是难题



首创全自动内存泄露修复技术，并保证正确性
基于数据流分析全自动修复约30%的内存泄露
研究成果发表于ICSE'15

其他成果：浮点误差测试技术

误差常常导致灾难性后果



全自动查找程序中的浮点误差
发现GSL科学计算库中的多处潜在误差问题
相关研究成果发表于ICSE'15，并入围ACM SIGSOFT Distinguished Paper Award候选

期待与百度各位开展合作！