Locating Crashing Faults based on Crash Stack Traces

Liang Gong

For more details view the technical report
(publicly available. Search on Google)

软件崩溃缺陷定位方法的研究

2011开题答辩

Presented by

贡 亮 (2010212412)
导 师： 张洪宇

2011 Dissertation Presentation

提 纲

Introduction & framework

- **选题动机**
- 困难与挑战

- **选题背景**
- 软件崩溃报告机制
- 基于频谱的缺陷定位技术

- **研究内容**
- 研究目标
- 研究方案
- 研究数据
- 研究计划

Presented by

GL

贡 亮 (2010212412)
Gong. Liang

“Testing can only prove the presence of bugs, not their absence.”

Edsger W. Dijkstra

Windows

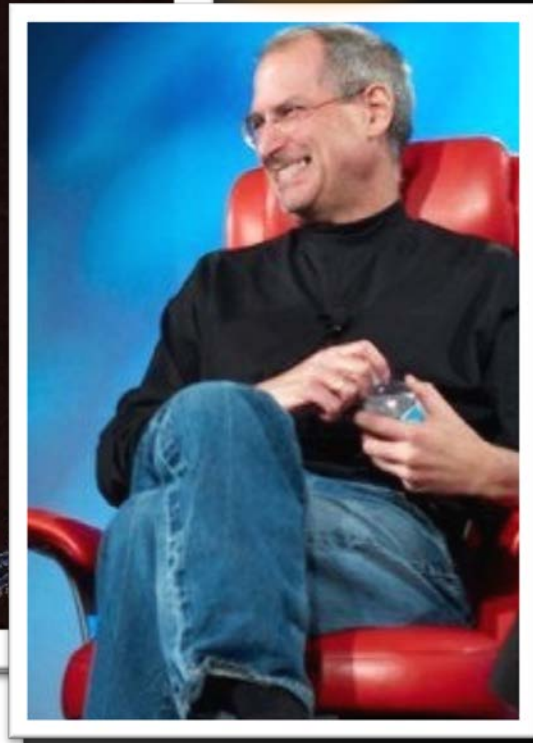
A fatal exception 0E has occurred at 0028:C0011E36 in VXD VMM(01) +
00010E36. The current application will be terminated.

* Press any key to terminate the current application.
* Press CTRL+ALT+DEL again to restart your computer. You will
  lose any unsaved information in all your applications.

Press any key to continue _

软件崩溃无处不在

如果不能及时修复软件
缺陷意味着客户的流失



当Windows发布的时候

当华尔奎这么做的时候

当坐地铁的时候

2011 Dissertation Presentation

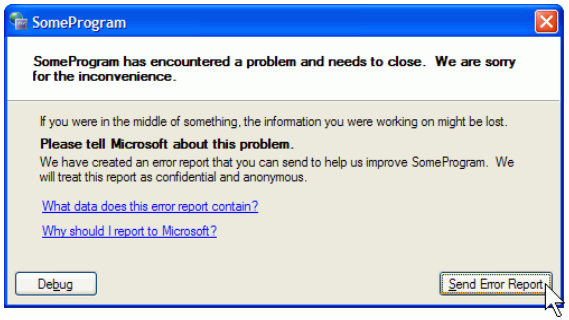
崩溃报告机制

Crash Reporting System

GNOME™

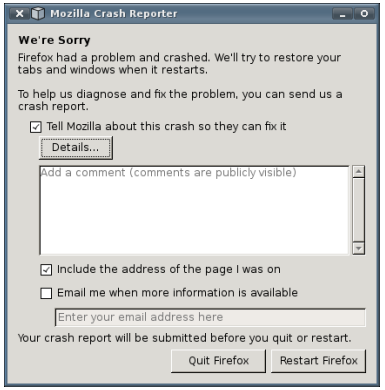
Bug Buddy

Microsoft®



Windows Error Reporting

mozilla FOUNDATION



Mozilla Crash Reporter

Presented by

贡亮 (2010212412)

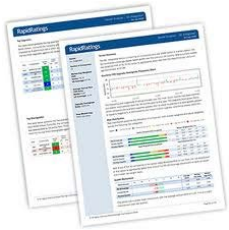
Gong. Liang

2011 Dissertation Presentation

崩溃报告机制

Crash Reporting System

软件崩溃



发送崩溃报告

崩溃报告收集

报告分类

同一种缺陷的崩溃信息

软件调试

Presented by

GL

贡亮 (2010212412)

Gong. Liang

选题动机

Manual Debugging

然而软件调试(***Debugging***)往往非常费时费力，通常开发人员被指定修复一个缺陷(***Fault***)之后需要阅读大量的崩溃报告(***Crash Report***) ,讨论，甚至重现崩溃场景(***Reproduce***)，推断出崩溃的具体位置然后修复缺陷。

Presented by

GL

贡亮 (2010212412)
Gong. Liang

2011 Dissertation
Presentation

选题动机

Manual Debugging

如果能够有某种技术自动定位缺陷那么将会提高缺陷定位效率，提升软件质量，降低维护成本。

Presented by

GL

贡亮 (2010212412)
Gong. Liang

2011 Dissertation Presentation

缺陷定位

Fault Localization

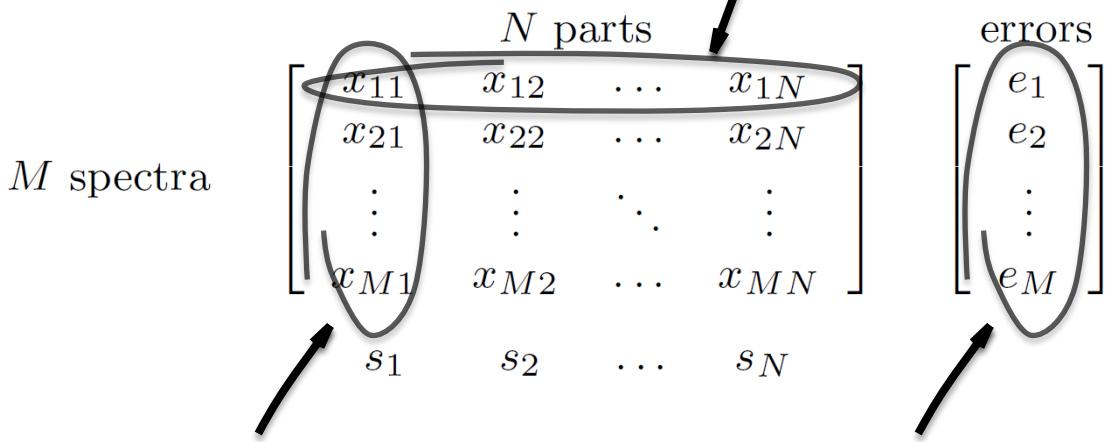
程序切片
(***Program Slicing***)

增量调试
(***Delta Debugging***)

频谱缺陷定位技术
(***Spectrum-base Fault Localization***)

程序频谱(***Program Spectra***)

每行表示次程序执行路径



每列表示一个语句在所有路径中执行次数

每行表示执行是否出错

Presented by

贡亮 (2010212412)
Gong. Liang

Fault Localization Ranking Formulas

$$\frac{a_{ef}}{a_{ef}+a_{nf}+a_{ep}}$$
$$\frac{a_{ef}}{a_{ef}+2\cdot(a_{nf}+a_{ep})}$$
$$\frac{a_{ef}}{a_{ef}+a_{nf}+a_{ep}}$$
$$\frac{2\cdot a_{ef}}{a_{ef}+a_{nf}+a_{ep}}$$
$$\frac{a_{ef}}{a_{nf}+a_{ep}}$$
$$\frac{1}{2}\cdot\left(\frac{a_{ef}}{a_{ef}+a_{nf}}+\frac{a_{ef}}{a_{ef}+a_{ef}}\right)$$
$$\frac{a_{ef}}{a_{ef}+a_{nf}+a_{ep}+a_{np}}$$
$$\frac{a_{ef}+a_{np}-a_{nf}-a_{ep}}{a_{ef}+a_{nf}+a_{ep}+a_{np}}$$
$$\frac{a_{ef}+a_{np}}{a_{ef}+a_{nf}+a_{ep}+a_{np}}$$
$$\frac{2\cdot(a_{ef}+a_{np})}{2\cdot a_{ef}+2\cdot a_{np}+a_{nf}+a_{ep}}$$
$$\frac{a_{ef}+a_{np}}{a_{nf}+a_{ep}}$$
$$\frac{a_{ef}}{a_{ef}+a_{np}+2\cdot a_{nf}+2\cdot a_{ep}}$$
$$\frac{a_{ef}+a_{np}}{a_{ef}+a_{nf}+2\cdot a_{nf}+2\cdot a_{ep}}$$
$$\frac{2\cdot a_{ef}-a_{nf}-a_{ep}}{2\cdot a_{ef}+a_{nf}+a_{ep}}$$
$$a_{cf}+a_{np}$$
$$\sqrt{a_{cf}+a_{np}}$$
$$\frac{a_{ef}}{\sqrt{(a_{ef}+a_{nf})\cdot(a_{ef}+a_{ep})}}$$
$$\frac{a_{ef}}{\min(a_{ef},a_{nf},a_{ep})}$$
$$\frac{a_{ef}+a_{nf}}{a_{ef}+a_{nf}+a_{ep}+a_{np}}$$
$$\frac{a_{ef}}{a_{ef}+a_{nf}+a_{ep}+\frac{10000\cdot a_{nf}\cdot a_{ep}}{a_{ef}}}$$
$$\left|\frac{a_{ef}}{a_{ef}+a_{nf}}-\frac{a_{ep}}{a_{ep}+a_{np}}\right|$$
$$a_{cf}$$
$$a_{cf}-a_{cp}$$
$$a_{ef}-\begin{cases}a_{ep} & a_{ep}\le 2\\ 2+0.1\cdot(a_{ep}-2) & 2\le a_{ep}\le 10\\ 2.8+0.001\cdot(a_{ep}-10) & a_{ep}\ge 10\end{cases}$$
$$\frac{a_{ef}\cdot a_{np}}{\sqrt{(a_{ef}+a_{ep})\cdot(a_{np}+a_{nf})\cdot(a_{ef}+a_{nf})\cdot(a_{ep}+a_{np})}}$$
$$\frac{a_{ef}\cdot a_{np}-a_{nf}\cdot a_{ep}}{\sqrt{(a_{ef}+a_{ep})\cdot(a_{np}+a_{nf})\cdot(a_{ef}+a_{nf})\cdot(a_{ep}+a_{np})}}$$
$$\frac{\frac{a_{ef}\cdot a_{np}}{\sqrt{(a_{ef}+a_{ep})\cdot(a_{np}+a_{nf})\cdot(a_{ef}+a_{nf})\cdot(a_{ep}+a_{np})}}}{\frac{a_{ef}\cdot a_{np}-a_{nf}\cdot a_{ep}}{\sqrt{(a_{ef}+a_{ep})\cdot(a_{np}+a_{nf})\cdot(a_{ef}+a_{nf})\cdot(a_{ep}+a_{np})}}}$$
$$\frac{2\cdot a_{ef}\cdot a_{np}-2\cdot a_{nf}\cdot a_{ep}}{(a_{ef}+a_{ep})\cdot(a_{np}+a_{nf})+(a_{ef}+a_{nf})\cdot(a_{ep}+a_{np})}$$
$$\frac{2\cdot a_{ef}\cdot a_{np}-2\cdot a_{nf}\cdot a_{ep}}{(a_{ef}+a_{ep})\cdot(a_{np}+a_{nf})+(a_{ef}+a_{nf})\cdot(a_{ep}+a_{np})}$$
$$\frac{4\cdot a_{ef}\cdot a_{np}-4\cdot a_{nf}\cdot a_{ep}-(a_{nf}-a_{ep})^2}{(2\cdot a_{ef}+a_{nf}+a_{ep})\cdot(2\cdot a_{np}+a_{nf}+a_{ep})}$$
$$\frac{4\cdot a_{ef}\cdot a_{np}-4\cdot a_{nf}\cdot a_{ep}-(a_{nf}-a_{ep})^2}{(2\cdot a_{ef}+a_{nf}+a_{ep})+(2\cdot a_{np}+a_{nf}+a_{ep})}$$
$$\frac{1}{2}\cdot\left(\frac{a_{ef}}{2\cdot a_{ef}+a_{nf}+a_{ep}}+\frac{a_{np}}{2\cdot a_{np}+a_{nf}+a_{ep}}\right)$$
$$\frac{1}{4}\cdot\left(\frac{a_{ef}}{a_{ef}+a_{ep}}+\frac{a_{ef}}{a_{ef}+a_{nf}}+\frac{a_{np}}{a_{np}+a_{ep}}+\frac{a_{np}}{a_{np}+a_{nf}}\right)$$

- 根据公式计算程序实体含有缺陷的可疑度

	L_1		L_i	L_{i+1}	L_{i+2}	L_{i+3}		L_n	P/F
T_1	9		2	2	0	1		9	pass
T_2	5		2	2	1	1		5	fail
T_3	13		2	2	1	1		13	pass
T_4	25		2	8	2	9		25	pass
T_5	25	...	2	5	3	4	...	25	pass
T_6	21		2	6	1	5		21	pass
T_7	15		2	4	0	3		15	fail
T_8	25		2	4	0	3		25	fail
T_9	18		2	5	1	4		18	pass
T_{10}	20		2	5	1	4		20	pass
	S_1		S_i	S_{i+1}	S_{i+2}	S_{i+3}		S_n	

2011 Dissertation Presentation

缺陷定位

Fault Localization

Ochiai

$$\frac{a_{ef}}{\sqrt{(a_{ef}+a_{nf})\cdot(a_{ef}+a_{ep})}}$$

Jaccard

$$\frac{a_{ef}}{a_{ef}+a_{nf}+a_{ep}}$$

Tarantula

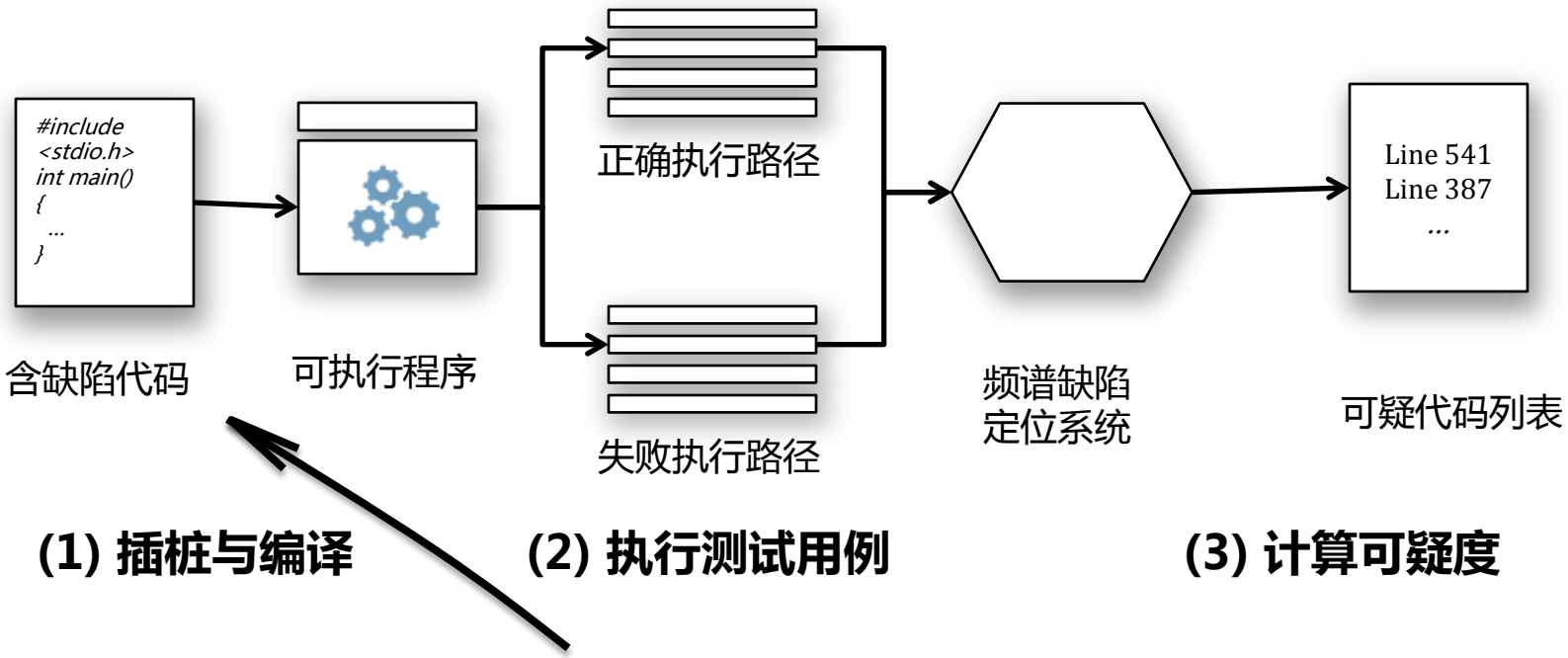
$$\frac{\frac{a_{ef}}{a_{ef}+a_{nf}}}{\frac{a_{ef}}{a_{ef}+a_{nf}}+\frac{a_{ep}}{a_{ep}+a_{np}}}$$

Presented by

GL 贡亮 (2010212412)
Gong. Liang

现有频谱缺陷定位技术框架

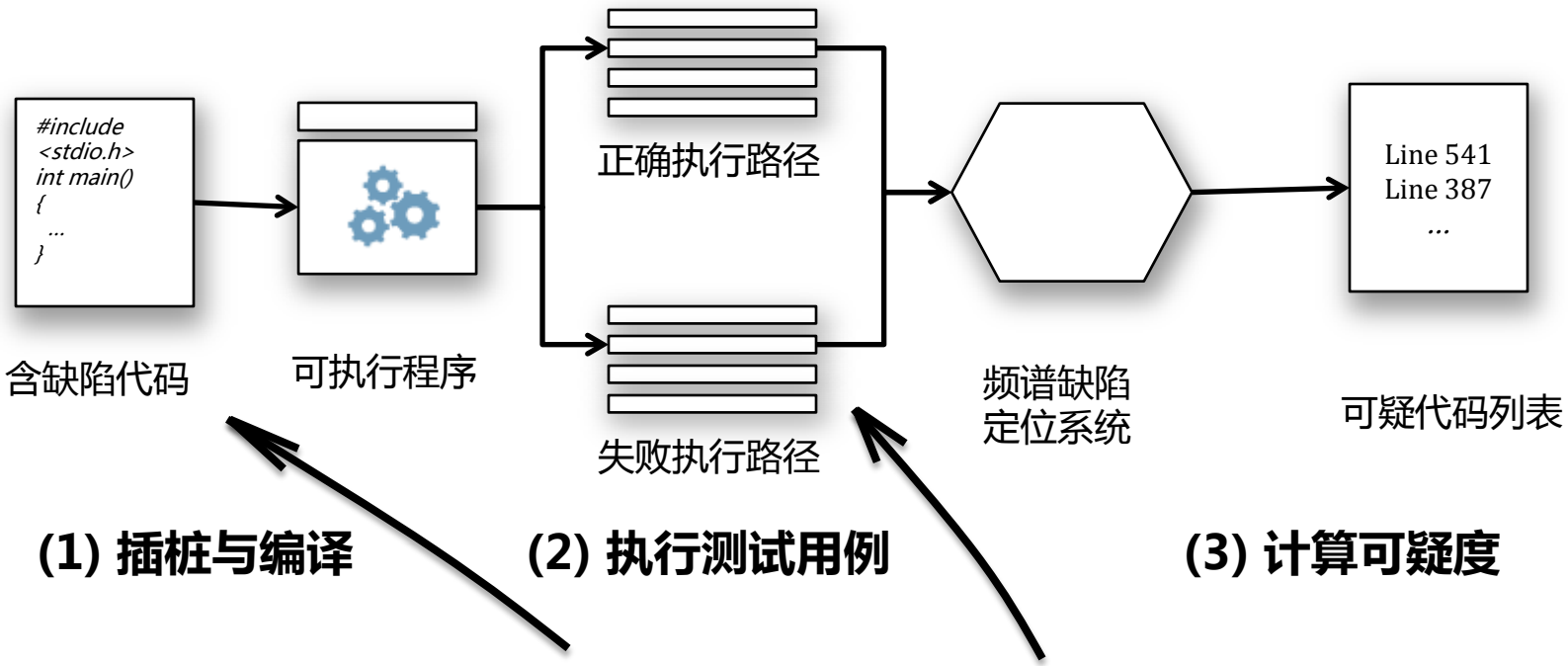
Conventional Fault Localization Framework



***Ben Liblit* 2003 年提出一种抽样插桩的方法，他们的方法在采样率为1/1000时可以保证插桩程序效率下降不超过5%，但是条件是每种缺陷对应的崩溃报告至少需要2300条，只适用于用户极广的软件系统比如*Office Xp*.**

现有频谱缺陷定位技术框架

Conventional Fault Localization Framework

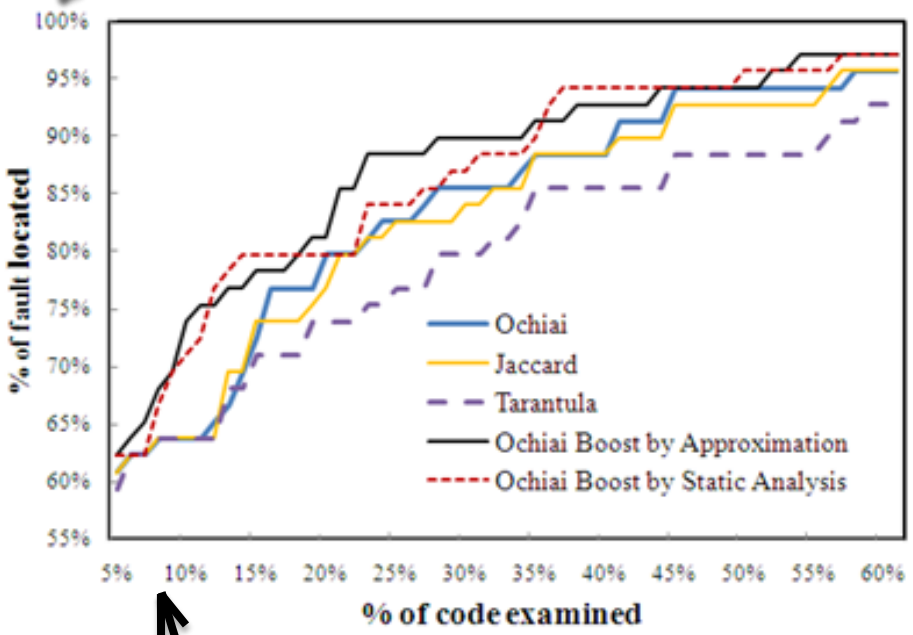


以往基于频谱缺陷定位的方法需要在客户端进行插桩，导致发布软件效率下降37%-625%导致用户体验变差，设置软件不可用

结果评估

Evaluation

纵轴表示定位缺陷百分比



(b) overall results in range of [5%, 60%]

横轴表示阅读代码百分比

Presented by

GL

贡亮 (2010212412)
Gong. Liang

崩溃报告

Crash Report

基于程序频谱(*Program Spectra*)的缺陷定位技术需要使用程序动态执行的路径信息。

*Gang-Ryung Uh*等人的研究发现：程序插桩会使程序变慢*37%-625%*，所以出于对客户体验的考虑，发布版本中的程序中不会进行插桩。

所以通常崩溃报告系统只会记录如下信息：

Information		Description
Details	*Product*	产品编号
	Date Processed	崩溃时间
	Signature	崩溃的函数
	Version	产品版本
	OS Version	操作系统版本
Modules		崩溃时所载入模块 以及对应模块版本
Raw Dump		软件崩溃时 所有线程的调用栈
Extensions		所有安装的插件 以及对应版本号
Comments		用户评价以及其他信息

Presented by

GL 贡亮 (2010212412)
Gong. Liang

问题

Current Situation

1

崩溃报告源源不断发来，
信息不能有效利用

- **Firefox** 每周收到至少12万条崩溃报告。
- **Microsoft** 已经收到了累计超过10亿条崩溃报告。

2

现有频谱缺陷定位技术需要崩溃执行路径，但是崩溃报告中没有。

$$\overbrace{\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1N} \\ x_{21} & x_{22} & \cdots & x_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{M1} & x_{M2} & \cdots & x_{MN} \end{bmatrix}}^{N \text{ parts}} \quad \overbrace{\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_M \end{bmatrix}}^{\text{errors}}$$

$$\begin{matrix} s_1 & s_2 & \cdots & s_N \end{matrix}$$

需要一种技术将频谱缺陷定位应用于崩溃报告

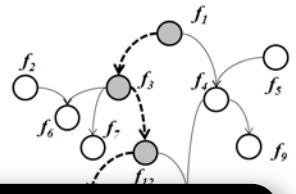
Presented by

GL

贡亮 (2010212412)
Gong. Liang

本研究提出的缺陷定位技术框架

New Fault Localization Framework



Frame 0	f_{11}
Frame 1	f_{12}
Frame 2	f_3
Frame 3	f_1

(B) 崩溃调用栈

$f_1 \rightarrow f_3$	$f_4 \rightarrow f_9$
$f_1 \rightarrow f_4$	$f_4 \rightarrow f_{13}$
$f_2 \rightarrow f_6$	$f_5 \rightarrow f_4$
$f_3 \rightarrow f_6$	$f_{10} \rightarrow f_{14}$
$f_3 \rightarrow f_7$	$f_{11} \rightarrow f_{14}$
$f_3 \rightarrow f_{12}$	$f_{11} \rightarrow f_{15}$
$f_{12} \rightarrow f_{11}$	$f_{12} \rightarrow f_{13}$

(C) 函数调用对

客户端程序不需要插桩，只保留调用栈等信息。

客户端

发布程序

程序崩溃

崩溃报告

发送至服务器

扩展出失败执行路径

距离	函数 ID
0	f_1, f_3, f_{12}, f_{11}
1	$f_4, f_6, f_7, f_{13}, f_{14}, f_{15}$
2	f_9

(D) 函数与调用栈距离

(E) 调用栈扩展

静态分析

函数调用关系

```
#include
<stdio.h>
int main()
{
  ...
}
```

含缺陷代码

插桩与编译

插桩程序

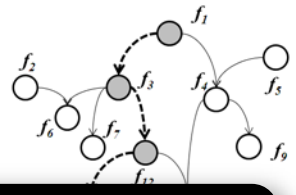
正确执行路径

基于频谱的
缺陷定位系统

可疑函数列表

本研究提出的缺陷定位技术框架

New Fault Localization Framework



(在函数 f_{11} 处崩溃)

Frame 0	f_{11}
Frame 1	f_{12}
Frame 2	f_3
Frame 3	f_1

(B) 崩溃调用栈

$f_1 \rightarrow f_3$	$f_4 \rightarrow f_9$
$f_1 \rightarrow f_4$	$f_4 \rightarrow f_{13}$
$f_2 \rightarrow f_6$	$f_5 \rightarrow f_4$
$f_3 \rightarrow f_6$	$f_{10} \rightarrow f_{14}$
$f_3 \rightarrow f_7$	$f_{11} \rightarrow f_{14}$
$f_3 \rightarrow f_{12}$	$f_{11} \rightarrow f_{15}$
$f_{12} \rightarrow f_{11}$	$f_{12} \rightarrow f_{13}$

(C) 函数调用对

客户端程序不需要插桩，只保留调用栈等信息。

客户端

程序崩溃

发布程序

崩溃报告

发送至服务器

扩展出失败执行路径

距离	函数 *ID*
0	f_1, f_3, f_{12}, f_{11}
1	$f_4, f_6, f_7, f_{13}, f_{14}, f_{15}$
2	f_9

(D) 函数与调用栈距离

f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
1	0	1	0	0	0	0	0	0	1	1	0	0	0
第一层扩展													
1	0	1	1	0	1	1	0	0	1	1	1	1	1
第二层扩展													
1	0	1	1	0	1	1	1	0	1	1	1	1	1

(E) 调用栈扩展

静态分析

函数调用关系

<stdio.h>
int main()

含缺陷代码

插桩与编译

插桩程序

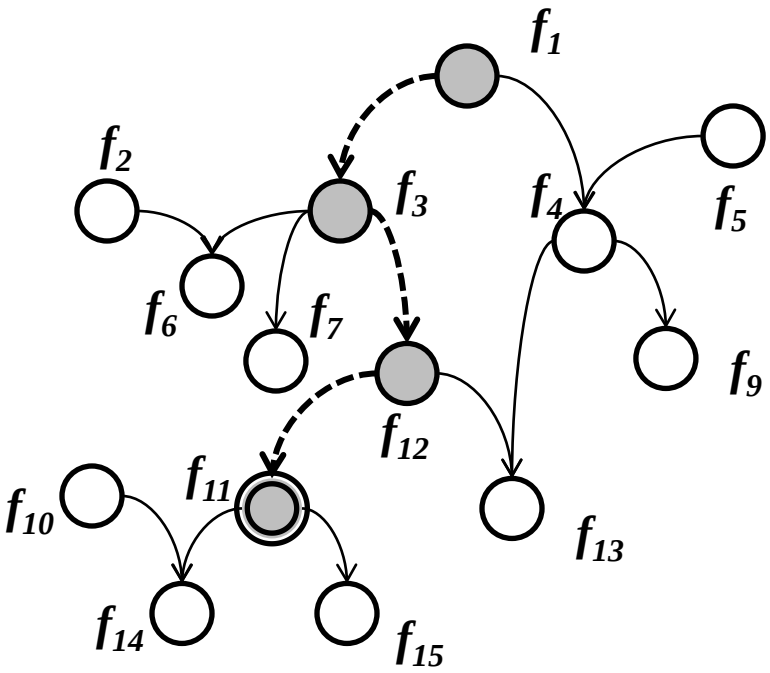
正确执行路径

基于频谱的缺陷定位系统

可疑函数列表

调用栈扩展

Illustration



(A) 函数调用图

(在函数 f_{11} 处崩溃)

Frame 0	f_{11}
Frame 1	f_{12}
Frame 2	f_3
Frame 3	f_1

(B) 崩溃调用栈

$f_1 \rightarrow f_3$	$f_4 \rightarrow f_9$
$f_1 \rightarrow f_4$	$f_4 \rightarrow f_{13}$
$f_2 \rightarrow f_6$	$f_5 \rightarrow f_4$
$f_3 \rightarrow f_6$	$f_{10} \rightarrow f_{14}$
$f_3 \rightarrow f_7$	$f_{11} \rightarrow f_{14}$
$f_3 \rightarrow f_{12}$	$f_{11} \rightarrow f_{15}$
$f_{12} \rightarrow f_{11}$	$f_{12} \rightarrow f_{13}$

(C) 函数调用对

Presented by

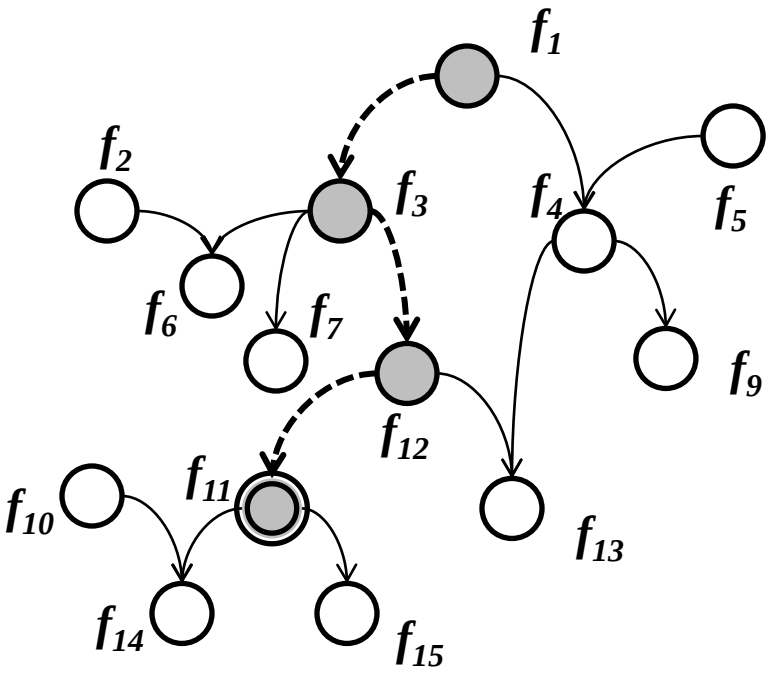
GL

贡亮 (2010212412)
Gong. Liang

2011 Dissertation Presentation

缺陷定位

Fault Localization



(A) 函数调用图

(在函数 f_{11} 处崩溃)

距离	函数 ID
0	f_1, f_3, f_{12}, f_{11}
1	$f_4, f_6, f_7, f_{13}, f_{14}, f_{15}$
2	f_9

(D) 函数与调用栈距离

f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
1	0	1	0	0	0	0	0	0	1	1	0	0	0
第一层扩展													
1	0	1	1	0	1	1	0	0	1	1	1	1	1
第二层扩展													
1	0	1	1	0	1	1	1	0	1	1	1	1	1

(E) 调用栈扩展

Presented by

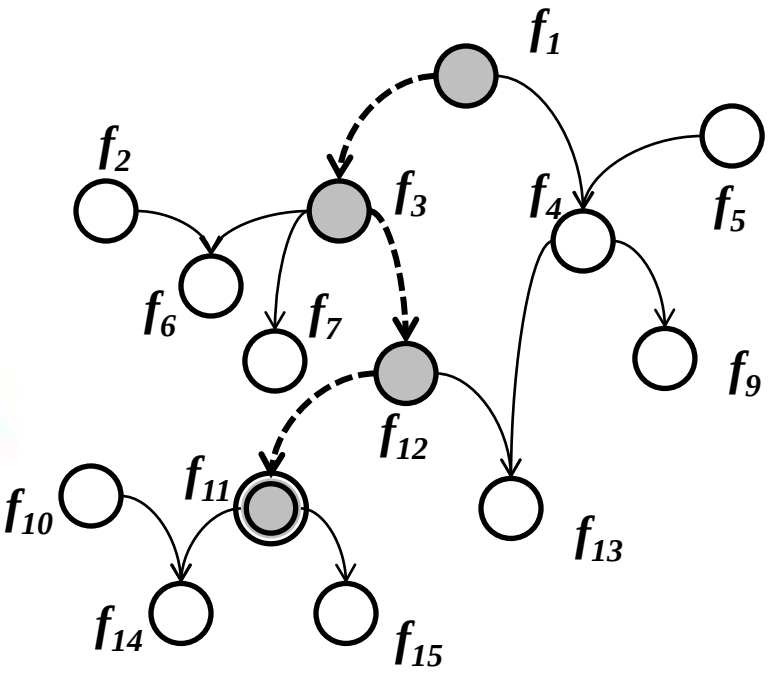
贡亮 (2010212412)

Gong. Liang

2011 Dissertation Presentation

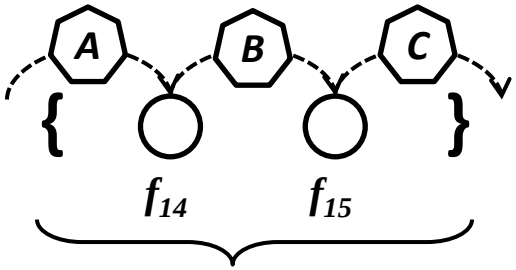
缺陷定位

Fault Localization



(A) 函数调用图

(在函数 f_{11} 处崩溃)



f_{11} 函数体中的循环

(F) f_{11} 扩展带来的噪音

Presented by

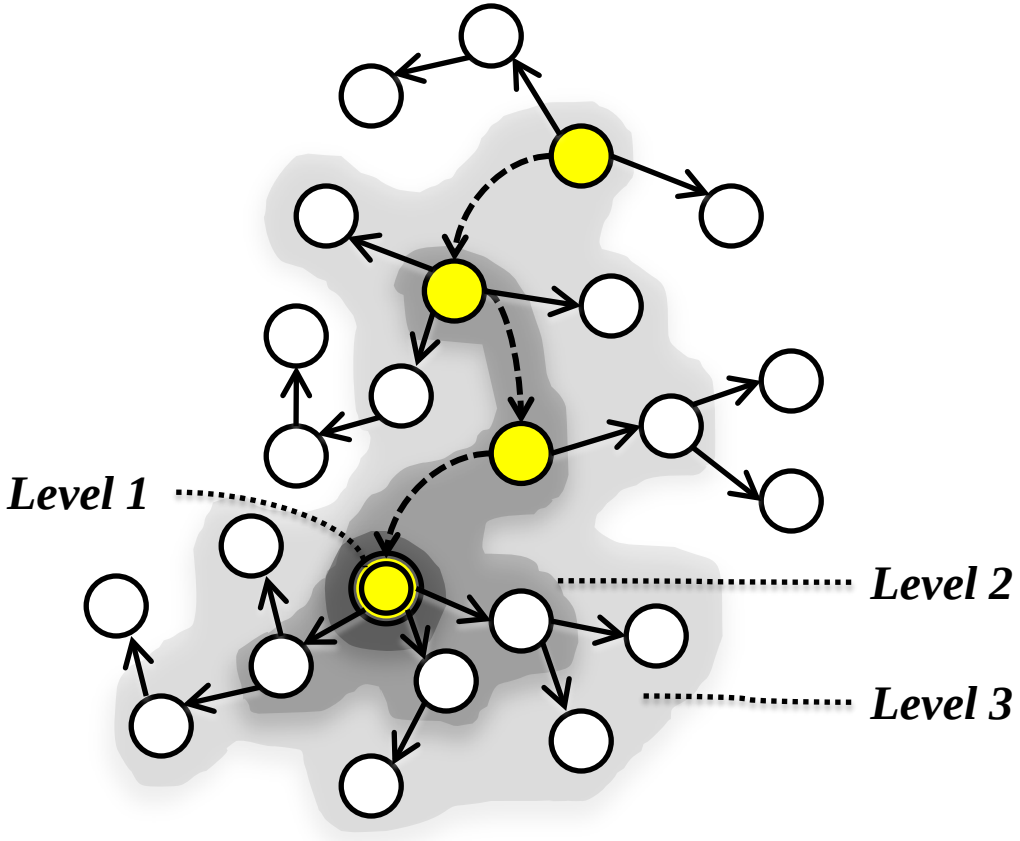
GL

贡 亮 (2010212412)

Gong. Liang

启发式规则

Heuristic-1



Presented by

GL

贡 亮 (2010212412)
Gong. Liang

Firefox

Mozilla Firefox is a free and open source web browser.
As of August 2011, Firefox is the second most widely used browser.

实验数据 Experimental Dataset

Fennec

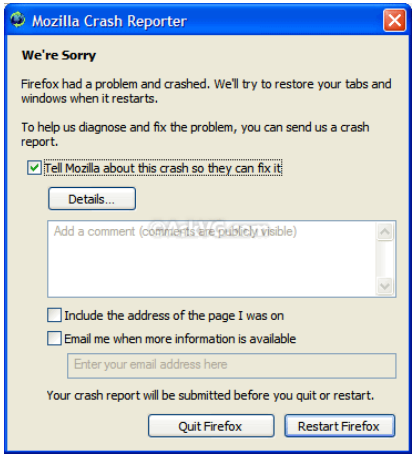
Firefox 3.6

Firefox 4.0

Thunderbird

SeaMonkey

Camino

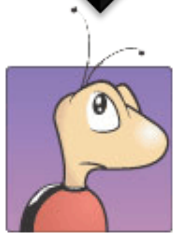


Breakpad

客户端采集崩溃报告

收集崩溃报告，统计信息
产品质量分析

Socorro

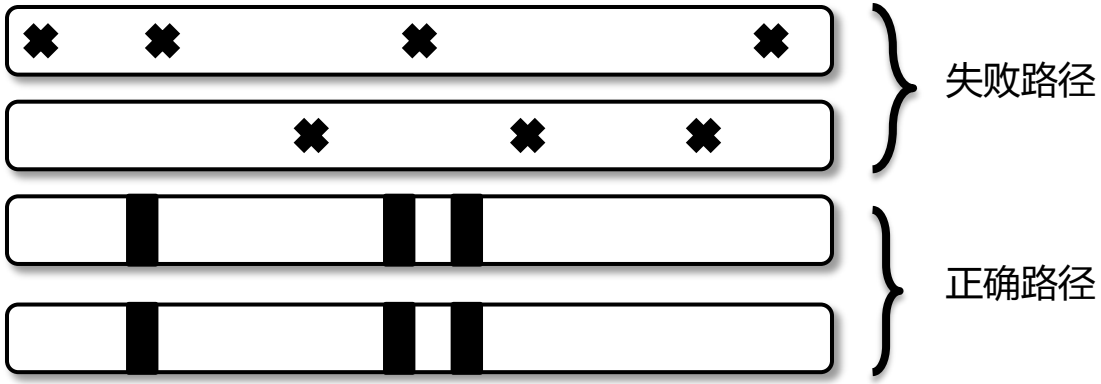


Bugzilla

对影响最大的崩溃人工建立
缺陷报告，指定开发人员调试

困难&解决方案
Difficulties and Solutions

合成频谱中的噪音与未被覆盖的路径



✖ 表示噪音

▮ 表示未被正确路径覆盖的函数

噪音问题：
•提出一些启发式方法预测可能的噪音并减低其相应的权值。

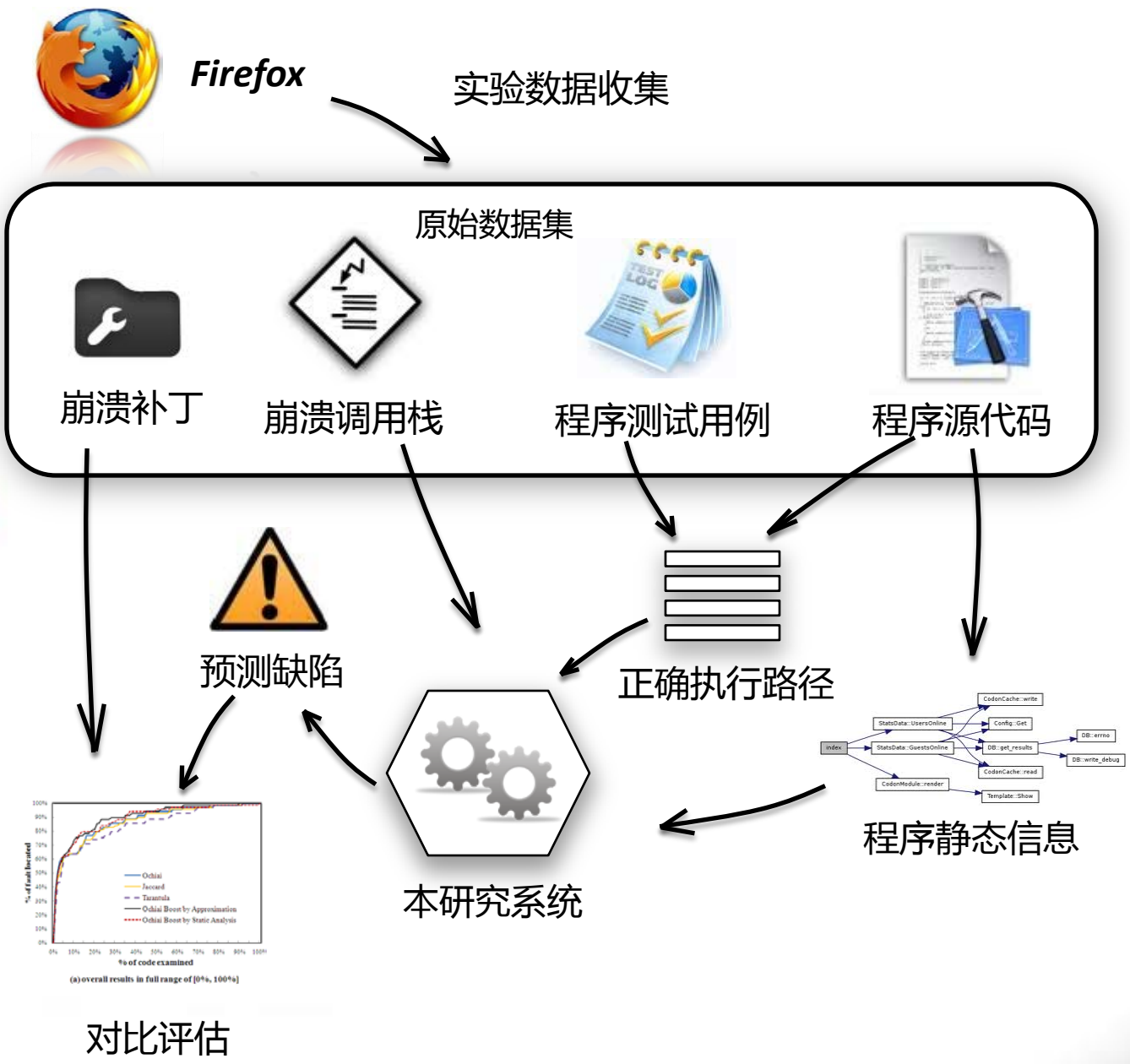
代码覆盖率问题：
•所有没有覆盖到的代码片段统一降低可疑度。
•对于所有没有覆盖到的代码片段 取平均值
•使用KNN算法找到最相似的近邻，求近邻平均值。

Presented by
GL 贡亮 (2010212412)
Gong. Liang

2011 Dissertation Presentation

实验流程

Evaluation Process



Presented by

贡亮 (2010212412)

Gong. Liang

This works is done in 2011.
For more details view the technical report:

Search on Google:
locating crashing faults based on crash stack traces

Thank you!