软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2013,24(7):1484−1494 [doi: 10.3724/SP.J.1001.2013.04294]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种基于信息量的缺陷定位方法*

丁 晖[1,2], 陈 林[1,2], 钱 巨[1,3], 许 蕾[1,2], 徐宝文[1,2]

[1](计算机软件新技术国家重点实验室(南京大学),江苏 南京 210093)

[2](南京大学 计算机科学与技术系,江苏 南京 210093)

[3](南京航空航天大学 计算机科学与技术学院,江苏 南京 210016)

通讯作者: 丁晖, E-mail: dinghui85@gmail.com, http://ise.nju.edu.cn

摘 要: 缺陷定位是软件调试过程中的重要环节,它通过利用程序信息和测试信息来定位软件中的错误.借助于事件信息量,提出一种基于事件信息量的缺陷定位方法——SIQ(suspiciousness based on information quantity).SIQ 方法根据测试信息中不同事件的类型及其发生的概率,结合语句的执行信息,动态计算和调整缺陷定位的结果.通过大量的实验分析和对比,SIQ 方法在多个数据集上表现出了很好的稳定性,与几种现有的缺陷定位方法相比,SIQ 方法的缺陷定位效果也更加准确.

关键词: 程序调试;缺陷定位;信息量

中图法分类号: TP311 **文献标识码**: A

中文引用格式: 丁晖,陈林,钱巨,许蕾,徐宝文.一种基于信息量的缺陷定位方法.软件学报,2013,24(7):1484−1494. http://www.jos.org.cn/1000-9825/4294.htm

英文引用格式: Ding H, Chen L, Qian J, Xu L, Xu BW. Fault localization method using information quantity. Ruan Jian Xue Bao/Journal of Software, 2013,24(7):1484−1494 (in Chinese). http://www.jos.org.cn/1000-9825/4294.htm

Fault Localization Method Using Information Quantity

DING Hui[1,2], CHEN Lin[1,2], QIAN Ju[1,3], XU Lei[1,2], XU Bao-Wen[1,2]

[1](State Key Laboratory for Novel Software Technology (Nanjing University), Nanjing 210093, China)

[2](Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China)

[3](College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)

Corresponding author: DING Hui, E-mail: dinghui85@gmail.com, http://ise.nju.edu.cn

Abstract: Fault localization is an important step in debugging. It makes use of the information from the source code and testing to locate bugs in software. This paper presents a method using the information quantity of testing events, called SIQ (suspiciousness based on information quantity). SIQ adjusts the fault localization procedure dynamically according to the testing events and the execution information of statements. In a series of experiments and analysis, SIQ takes on excellent stability in multiple data sets and has better localization accuracy compared to several existing methods.

Key words: debug; fault localization; information quantity

缺陷定位是软件调试过程中一件很困难的工作,对大规模软件更是如此.因此,缺陷定位多年来一直是软件工程研究领域中的一个热点问题.缺陷定位利用程序信息和测试信息找出错误语句或者预测错误语句可能存在的范围,辅助开发人员找到缺陷代码.

多年来,人们在缺陷定位的研究中提出了许多方法,主要通过程序的静态信息[1−4]和动态信息[5,6]来定位程

* 基金项目: 国家自然科学基金(90818027, 61003020, 61170071, 60903026); 国家重点基础研究发展计划(973)(2009CB320703); 江苏省自然科学基金(BK2011190)

收稿时间: 2011-09-27; 修改时间: 2012-03-19; 定稿时间: 2012-07-24

© 中国科学院软件研究所 http://www.jos.org.cn

序错误.获得静态信息的开销很大,对于大型软件,全面的静态分析甚至是不可能的.动态信息的收集只需要运行测试用例,并不会给测试带来过多的开销.同时,由于动态信息包含了程序运行时的信息,与利用静态信息的方法相比,可以提供接近甚至更准确的结果.针对利用动态信息的缺陷定位过程,郝丹等人提出了考虑用例相似度的缺陷定位方法,在保证定位效果的同时,大幅度减少了需要使用的用例数量,减轻了测试预言需要的工作[7,8].张震宇等人利用程序分支条件信息结合语句执行序列进行缺陷定位,对分支逻辑类型的错误有很好的定位效果[9].

程序频谱是一种表示程序运行时覆盖情况的信息.利用程序频谱信息进行缺陷定位,是目前比较切实有效的方法.然而,现有的基于频谱的方法由于利用的频谱信息不够充分,不能根据不同的测试用例集所得的数据动态调整算法,从而在不同测试集上的定位效果不够稳定,很难得到准确的结果.

此前,我们曾针对这种状况提出了一种利用广义边际效应的缺陷定位方法,一定程度上提高了缺陷定位的效果[10].在此基础上,考虑到应该充分收集和组合利用多种程序及测试信息,本文提出了一种基于事件信息量的缺陷定位方法.在西门子程序集和 space 程序上的实验分析表明,这种方法在不同测试集上具有良好的稳定性,且定位效果也优于现有的方法.

1 存在问题分析

有一定软件测试经验的人一般都有这样的体会:被越多失败测试用例执行过的代码越有可能出错;反之,被越多成功测试用例执行过的代码出错的可能越小.基于程序频谱的缺陷定位方法就基于这一朴素的经验.

程序频谱是一个二进制序列,每一位都对应表示程序中某个相应的成分(例如语句、分支、基本块、函数调用等),其取值表示对应的程序成分在某个用例中是否被执行.基于频谱的方法收集程序运行数据,计算每条语句可能出错的程度,称为可疑度.可疑度值越高,表明该语句出错的可能性越大.由于频谱信息较易收集,通常只需要语句级的插桩即可实现,输出结果也比较直观,且定位效果不错,因此得到了广泛的关注.人们已经提出了很多种基于频谱的缺陷定位方法[11]:Jones 等人首先提出 Tarantula 方法[5],取得了不错的缺陷定位效果;Abreu 等人使用了分子生物学中的公式计算语句与出错结果的相似因子,提出 Ochiai 方法[12,13],效果相比 Tarantula 方法有了一定的提升;Gonzalez 增加了排除正确语句的优化,提出了 Zoltar 方法[14];Naish 等人使用自组装映射中计算相似度的方法来寻找错误语句,引入 Kulczynski1 方法和 Kulczynski2 方法[11,15];Wong 等人认为,应该突出错误用例的影响,据此提出新的方法(以下简称 Wong 方法),定位效果相比 Tarantula 方法有了较大的提升[11].

Tarantula 方法采用了固定的参数来计算语句的可疑度,当测试用例数量变化,其中的成功用例或者失败用例数发生变化时,Tarantula 方法表现出一定的不稳定性,定位效果也受到影响.与 Tarantula 方法相比,Wong 方法重新考虑了正确执行的用例在可疑度计算公式中的权重,认为当正确执行的用例数量较多时,其在可疑度计算公式中的作用应当被削减.对于一条语句,当其正确执行的用例较少时(Wong 方法中设置为 2),Wong 方法与 Tarantula 方法有相同的排序结果;当正确执行用例的数量继续增加时,其影响将会减小.在单错误程序中,Wong 方法比 Tarantula 方法有更好的效果[6,11].

通过对多个程序及测试用例集的考察可以发现,在实际情况中,用例数据的分布变化是很大的.我们发现,测试用例集数据分布情况会对缺陷定位方法造成影响,而且这种影响有时候会很大.现有的很多方法总是在某些类型的用例集上表现得比较好,而在另外一些用例集上定位效果却不理想,缺陷定位方法表现出一定的不稳定性.我们希望,缺陷定位方法不仅有更好的平均效果,同时能够适应不同类型的数据集,尽可能地降低用例集对定位效果的影响,具有更好的稳定性.Wong 方法添加了对正确用例权重的考虑,效果得到了提升.然而这仍然不够,因为 Wong 方法是采用固定的数值来分段计算公式,当用例数量和其中结果发生变化时,固定的数值很难适应;并且, Wong 方法只考虑了语句执行时对可疑度的影响,语句未执行时用例的结果对缺陷定位也是有用的信息,尤其是对于分支条件错误以及多错误程序.对分支条件类的错误,当错误仅发生在判断边界的一个小范围内时,并不是每次执行都会引发错误,因此,语句未执行时的信息对判断此类错误就很必要了;对于多错误程序,错误之间的相互关联和作用,要充分利用收集的信息才能处理好这些关系.

© 中国科学院软件研究所 http://www.jos.org.cn

因此,如果能够充分利用收集的信息,并且能够根据数据分布情况动态调整不同参数的权重,可以预见,定位的效果会更稳定,准确度也会有所提高.那么,如何合理地组合收集的测试信息,如何动态调整参数权重呢?为了解决这些问题,本文提出了基于事件信息量的缺陷定位方法.

2 基于信息量的缺陷定位方法

为了提高缺陷定位的准确性,应该更充分地利用测试收集的信息.同时,为了让缺陷定位效果更加稳定,需要算法能够根据数据自动调整.因此,本文从两方面对缺陷定位算法加以改进:充分、合理地利用收集的信息以及根据测试数据动态调整算法.

2.1 对频谱参数含义的分析

在检查代码的过程中,人们通常习惯于把语句作为基本的单位,因此我们取语句作为频谱中每一位对应的基本成分.当一组用例执行之后,对每个语句 s 可以得到一组这样的二进制序列,分别对应每个用例中的语句执行情况.根据这些序列,对每条语句 s 可以计算出一组变量数值$\langle a_{np},a_{nf},a_{ep},a_{ef}\rangle$,$a_{np}$ 表示语句没有执行且测试结果正确的次数,a_{nf} 表示语句没有执行且测试结果错误的次数,a_{ep} 表示语句被执行且测试结果正确的次数,a_{ef} 表示语句被执行且测试结果错误的次数.

通过分析频谱中各参数的含义,可以得到每个参数对其对应语句可疑度的影响趋势.a_{ef} 的数值高,说明语句的执行倾向于使结果出错;a_{np} 这个参数很少被现有的缺陷定位方法利用,但这个数值对分支条件错误和多错误定位有一定的作用,该项数值增大,说明语句未执行趋向于使结果正确.因此,a_{ef} 和 a_{np} 的值越大,语句可疑度越高. a_{ep} 的数值增大,说明语句执行且用例正确的次数多,语句更倾向于是正确的.a_{nf} 表示语句未执行且结果出错的次数.a_{ep} 和 a_{nf} 的值越大,语句的可疑度越低.

2.2 测试事件的信息量

现有的缺陷定位方法利用频谱中参数对语句可疑度的影响趋势进行缺陷定位.然而在大量的实验中我们发现:频谱数据在不同测试用例集上的分布情况差别很大,现有的方法在不同数据集上很难得到稳定的结果.因此,为了适应不同的测试用例集,仅仅知道频谱中参数对可疑度计算的影响趋势还不够,还需要量化这些影响.

实际上,频谱中的每个参数都对应了测试中事件的组合,如 a_{ef} 对应于“语句执行”和“用例结果错误”这两个事件的组合,而参数本身的数值对应于这个事件组合发生的次数.现有的方法只利用了这些事件发生的次数,并没有考虑这些事件组合本身的信息.

要找到程序中的错误,需要关注测试中的异常情况.例如,在大量的测试中,仅有少量的用例失败,通过对少量错误用例的观察可以更快地寻找到错误;反之,若大部分用例都失败了,则观察少量通过的用例可以获得更多的信息.因此,对于缺陷定位,测试中的事件发生的概率越大,其所包含的信息越少,每一个这样的事件对缺陷定位的作用越小;反之,发生概率低的事件包含的信息多,每一个事件对缺陷定位的作用更大.这与信息论中信息量的概率相合,为此,我们引用信息量的概念.信息量(information quantity)是事件包含信息多少的度量.信息是用不确定性的量度定义的:事件出现的概率越小,其信息量就越大;事件出现的概率越大,其信息量就越小.

在频谱信息中,测试中发生的事件都是离散的事件.根据信息论中的定义:对于离散事件 x,若其发生概率为 $P(x)$,则 x 的信息量 I_x 为

$$I_x=\log\frac{1}{P(x)}=-\log P(x).$$

这样,频谱中参数对可疑度计算的影响就可以由其相关事件的信息量来调整.

2.3 考虑事件信息量的可疑度计算公式

我们首先计算用例执行结果所包含的信息量.在测试用例集中,设执行结果正确和错误的用例数分别为 p 和 f.假设 p 或 f 为 0,表示用例执行结果全部正确或全部错误,此时,只有语句的执行信息能够使用,语句会被分为执行和未执行两类,所有在测试过程中执行过的语句可疑度相同,无法利用这样的信息来进行缺陷定位.因此,p

© 中国科学院软件研究所 http://www.jos.org.cn

和 f 应该都非 0.根据离散事件信息量的定义,事件 p 和 f 所包含的信息量分别为

$$I_f = -\log P(f) = -\log\frac{f}{p+f},$$

$$I_p = -\log P(p) = -\log\frac{p}{p+f}.$$

据此,就可接着再计算语句执行情况对应的信息量.对语句 s,设其执行与未执行的次数分别为 $e(s)$和 $n(s)$.$e(s)$和 $n(s)$可能为 0,对应语句总是执行或者总是不执行.这种边界情况会使事件信息量的计算出现问题.为了解决这种问题,我们可以取近似的情况计算信息量.当概率为 0 时,事件不发生,此时取发生 1 次代替;当概率为 1 时,事件总是发生,此时取只有 1 次不发生代替.当用例数量达到一定程度时,这种近似使事件信息量的计算更平滑,并且针对本文提出的可疑度计算公式(1),这种近似结果在边界情况下会被作为因子消去,对计算并不产生影响.

对于事件 k,k_s 表示语句 s 发生的事件 k,其发生概率为 $P(k_s)$,定义 $I_k(s)$为

$$I_k(s)=\begin{cases} I(k_s), & \text{if } 0<P(k_s)<1 \\ -\log\dfrac{1}{e(s)+n(s)}, & \text{if } P(k_s)=0 \\ -\log\left(1-\dfrac{1}{e(s)+n(s)}\right), & \text{if } P(k_s)=1 \end{cases}.$$

这里,事件 k 为语句执行(e)或语句未执行(n).$I_e(s)$表示“语句 s 执行”这个事件所包含的信息量,$I_n(s)$表示“语句 s 未执行”所包含的信息量,表示如下:

$$I_e(s) = -\log P(k_e) = -\log\frac{e(s)}{e(s)+n(s)},$$

$$I_n(s) = -\log P(k_n) = -\log\frac{n(s)}{e(s)+n(s)}.$$

在前面的分析中已经得出:对语句 s,a_{ef} 和 a_{np} 越大,其可疑度越大;a_{ep} 和 a_{nf} 越大,其可疑度越小.同时,根据每个数值 a_{xy},其权重为对应的事件 x 和 y 信息量的乘积,当相关事件信息量较大时,参数的权重增大;反之亦然.据此,可得如下可疑度计算公式:

$$susp(s)=\frac{I_e(s)\cdot I_f\cdot a_{ef}+I_n(s)\cdot I_p\cdot a_{np}}{I_e(s)\cdot I_p\cdot a_{ep}+I_n(s)\cdot I_f\cdot a_{nf}} \tag{1}$$

本文称这种基于事件信息量的可疑度计算方法为 SIQ(suspiciousness based on information quantity,基于信息量的缺陷定位方法).

2.4 分 析

2.4.1 SIQ 方法对现有方法的解释

根据本文前面对频谱中参数的研究,我们得出:为了让缺陷定位结果更加稳定和准确,频谱中参数对可疑度的影响需要根据其对应事件的信息量动态地加以调整.当参数对应事件的信息量增大时,在 SIQ 方法中的权重提升;反之,则下降.为了进一步说明 SIQ 方法的性能,我们利用 SIQ 方法对一些缺陷定位方法(见表 1)进行了分析.

基于频谱的方法的结果给出语句的排序,可疑度的作用是确定语句排列的先后次序,因此,利用排序效果相同的可疑度计算公式进行缺陷定位的结果也是相同的.Naish 等人据此证明了 Tarantula 方法等价于[11]:

$$susp(s)=\frac{a_{ef}}{a_{ep}} \tag{2}$$

对于 SIQ,当忽略语句未执行时的信息时,可得

© 中国科学院软件研究所 http://www.jos.org.cn

$$susp(s)=\frac{I_f\cdot a_{ef}}{I_p\cdot a_{ep}} \tag{3}$$

对任一组用例,I_f 与 I_p 是定值,不难证明公式(2)与公式(3)的排序效果相同,即公式(3)与 Tarantula 等价.因此我们可以得出,忽略语句未执行信息时,SIQ 方法与 Tarantula 方法等价.

Table 1 Metrics of suspiciousness

表 1 可疑度计算公式

方法名称	可疑度计算公式
Tarantula	$susp(s)=\dfrac{\frac{a_{ef}}{a_{ef}+a_{nf}}}{\frac{a_{ef}}{a_{ef}+a_{nf}}+\frac{a_{ep}}{a_{ep}+a_{np}}}$
Ochiai	$susp(s)=\dfrac{a_{ef}}{\sqrt{(a_{ef}+a_{nf})(a_{ef}+a_{ep})}}$
Zoltar	$susp(s)=\dfrac{a_{ef}}{a_{ef}+a_{nf}+a_{ep}+\frac{10000a_{nf}a_{ep}}{a_{ef}}}$
Kulczynski1	$susp(s)=\dfrac{a_{ef}}{a_{nf}+a_{ep}}$
Kulczynski2	$susp(s)=\frac{1}{2}\left(\dfrac{a_{ef}}{a_{ef}+a_{nf}}+\dfrac{a_{ef}}{a_{ef}+a_{ep}}\right)$
Wong	$susp(s)=\begin{cases}a_{ef}-a_{ep}, & \text{if } a_{ep}\leqslant 2\\ a_{ef}-(2+0.1\times(a_{ep}-2)), & \text{if } 2<a_{ep}\leqslant 10\\ a_{ef}-(2.8+0.001\times(a_{ep}-10)), & \text{if } a_{ep}>10\end{cases}$
Wong1	$susp(s)=a_{ef}$
Wong2	$susp(s)=a_{ef}-a_{ep}$
TanDG	$susp(s)=\begin{cases}\dfrac{a_{ef}\times(1+a_{ef})}{2}\times\dfrac{a_{ep}+a_{np}}{a_{ef}+a_{nf}}-a_{ep}, & \text{if } a_{ef}\neq 0\\ a_{nf}-a_{ep}, & \text{if } a_{ef}=0\end{cases}$

SIQ 方法也可以用于解释 Wong 方法.在 Wong 方法中,当 a_{ep} 增大时,其权重下降[6].以 SIQ 的观点来解释,当 a_{ep} 增大时,事件 e 和 p 的信息量减少,对应的权重也会降低.Wong 方法采用了固定的参数,SIQ 方法可以更平滑地调整可疑度的计算.

Wong1 方法和 Wong2 方法是两种比较简单的计算公式,Wong1 方法只考虑了 a_{ef} 的值,没有利用其他信息;Wong2 利用 a_{ef} 和 a_{ep} 的绝对差值,没有考虑用例的数量等问题.在实际应用中,这两种方法的效果并不理想.

Kulczynski1 方法在 Tarantula 方法的基础上考虑了 a_{nf} 的作用.当 SIQ 方法忽略 a_{np} 并将所有信息量的权重看做相同时,与 Kulczynski1 方法等价.

Kulczynski2 方法是对 Kulczynski1 方法的改进.对 Kulczynski2 方法进行等价变化,可得:

$$susp(s)=\frac{1}{2}\left(\frac{a_{ef}}{a_{ef}+a_{nf}}+\frac{a_{ef}}{a_{ef}+a_{ep}}\right)=\frac{1}{2T}\times\left(\frac{1}{F\%}+\frac{1}{e\%}\right)\times a_{ef},$$

其中,T 是测试用例总数,F%是执行结果错误的概率,e%是语句 s 执行的概率.

对所有语句,T 为定值,故 Kulczynski2 方法的排序作用等价于:

$$susp'(s)=\left(\frac{1}{F\%}+\frac{1}{e\%}\right)\times a_{ef}.$$

因此,Kulczynski2 方法实际上是利用执行出错和语句执行的概率对 a_{ef} 的权重进行调整.SIQ 方法使用执行出错和语句执行这两个事件的信息量对 a_{ef} 的权重进行调整.Kulczynski2 方法只利用了 a_{ef} 参数进行定位,调整

© 中国科学院软件研究所 http://www.jos.org.cn

权重考虑的因素与 SIQ 方法相同,不过使用了事件概率倒数相加的方式,SIQ 方法利用的信息更加全面.同时,我们注意到:测试中,用例执行出错与语句执行这两个事件并不是独立的.因此,SIQ 方法并没有使用信息量相加的方法作为权重.

Ochiai 方法可以进行如下变换:

$$susp(s)=\frac{a_{ef}}{\sqrt{(a_{ef}+a_{nf})(a_{ef}+a_{ep})}}=\frac{a_{ef}}{\sqrt{F\cdot e}}=\frac{1}{T}\left(\frac{1}{\sqrt{F\%}}\times\frac{1}{\sqrt{e\%}}\right)a_{ef},$$

其中,T 是测试用例总数,F%是执行结果错误的概率,e%是语句 s 执行的概率.

可见,与 Kulczynski2 方法类似,Ochiai 方法也可以看做是利用语句执行出错与语句执行的概率对 a_{ef} 进行调整,进行缺陷定位.SIQ 方法不仅考虑了 a_{ef} 的调整,同时还综合考虑了其他几个参数的影响.

对 Zoltar 方法进行变换,可得

$$susp(s)=\frac{a_{ef}}{a_{ef}+a_{nf}+a_{ep}+\dfrac{10000a_{nf}a_{ep}}{a_{ef}}}=\frac{a_{ef}}{F+a_{ep}+\dfrac{10000a_{ep}}{a_{ef}}\cdot a_{nf}},$$

式中,F 表示错误用例总数,为定值.与 Kulczynski1 方法类似,Zoltar 方法在 Tarantula 的基础上加入了对 a_{nf} 的考虑.所不同的是,Zoltar 方法是给 a_{nf} 设置了非常大的权重,在错误用例所占比例较小的情况下有较好的效果[11,14];然而对于错误用例所占比例较大的程序,Zoltar 方法的效果并不理想.SIQ 方法不仅考虑了 a_{nf} 的权重,同时还调整了其他参数的权重以适应不同的用例集.

在之前的研究中,我们提出了一种利用广义边际效应的缺陷定位方法 TanDG.TanDG 方法通过增加错误用例的权重来提高缺陷定位效果.与 Zoltar 方法类似,TanDG 方法在错误用例较少的情况下效果较好;但当错误用例较多时,效果不够理想.SIQ 方法在错误用例较少时,用例执行错误的信息量 I_f 较大,增加了错误用例的权重,这时,SIQ 方法的思想与 TanDG 方法是一致的;当错误用例增多时,I_f 会下降,减弱了错误用例的影响,增加了正确用例的影响.通过这样的调整,SIQ 方法可以在不同的用例集上都表现出较为理想的效果.

2.4.2 边界情况

下面考虑两种边界的情况:

- 当 $n(s)$为 0 时,即语句 s 总是执行,公式(1)变化为

$$susp(s)=\frac{I_f\cdot a_{ef}}{I_p\cdot a_{ep}}=\frac{I_f\cdot f}{I_p\cdot p} \tag{4}$$

  此时,语句执行与否的信息已经无法对定位产生影响,语句的可疑度为所有错误执行所包含的信息量与所有正确执行所包含的信息量之比;

- 当 $e(s)$为 0 时,即语句没有执行,此时,公式(1)变化为

$$susp(s)=\frac{I_p\cdot a_{np}}{I_f\cdot a_{nf}}=\frac{I_p\cdot p}{I_f\cdot f} \tag{5}$$

  这种情况下,结果与公式(4)对称.对于没有执行的语句,其对缺陷的贡献与总是执行的语句互补,这符合直觉.

通过分析可以看出:SIQ 方法对频谱信息的使用更加充分.同时,基于信息量的参数自适应调整也更加平滑;并且,Tarantula 等方法也可以用 SIQ 方法的思想来解释.这样,SIQ 方法为这样一些现有的缺陷定位方法提供了一个统一的理论框架.

3 实验及结果分析

3.1 实验对象

为便于与现有方法的比较,实验采用目前使用广泛的西门子程序集和 space 程序[16].西门子程序集包含 7

© 中国科学院软件研究所 http://www.jos.org.cn

个程序,每个程序有一个正确版本和若干植入错误的版本.每个程序同时还配有一些测试用例以及用例集,用例集中的用例满足不同要求的代码覆盖能力,有几种类型的代码覆盖,见表 2.space 是一个完整的程序,其详细信息见表 3.不同类型的覆盖率对缺陷定位的效果是会产生影响的,后面的实验中将给出比较.程序的类型以及错误的类型对缺陷定位同样会产生影响,西门子程序集中 7 个程序的具体信息见表 3.

Table 2 Coverage types of the Siemens test suites

表 2 西门子用例集代码覆盖率类型

覆盖率类型	描述
testplans-cov	用例达到分支覆盖,并且无冗余
testplans-rand	根据 testplans-cov 中用例集所包含用例的平均数量 A,随机选取 $A/2$ 到 $2\times A$ 数量的用例
testplans-rand-covsize	按照 testplans-cov 中每个用例集中用例的数量,随机选取相同数量的用例构成每个用例集
testplans-bigcov	用例达到分支覆盖,但其中包含冗余
testplans-bigrand	按照 testplans-bigcov 中每个用例集中用例的数量,随机选取相同数量的用例构成每个用例集
universe	所有测试用例

Table 3 Information of the test programs

表 3 实验程序信息

程序名	错误版本数	代码行数	测试用例数	简要描述
print_tokens	7	344	4 130	词法分析程序
print_tokens2	10	355	4 115	词法分析程序
schedule	9	292	2 650	优先级调度
schedule2	10	262	2 710	优先级调度
replace	32	512	5 542	正则表达式替换
tcas	41	135	1 608	海拔高度分离
tot_info	23	273	1 052	信息度量
space	38	9 562	13 525	语言注释器

注:前面 7 个是西门子程序集中包含的程序,它们的规模相对较小,space 程序的规模更大.

在西门子和 space 的错误版本中:有些版本存在问题,未被使用;有些版本的错误会导致异常终止,这些错误版本是:print_tokens2 v10,replace v32,tcas v38,space 的 v25,v26 和 v30;还有的版本改动并未引发错误:包括 schedule2 v9 和 space 中的 v1,v2,v32,v34,也未被实验采用;有些版本的错误发生在头文件中,大多是一些宏定义或者变量值赋值,这些版本在一些文献中是未被使用的[6,11].为了更充分地利用西门子的程序集,本文的实验将这类错误定义在使用宏定义或者变量的地方,并不丢弃.因此,本实验采用西门子的 128 个版本和 space 的 31 个版本.

3.2 实验步骤和配置

为了更全面地反映各种方法的效果,实验运行了每个版本的所有用例集.这意味着仅对西门子的 128 个版本、每个版本包含 5 个类型的用例集,除 universe 外,其他每个类型大概包含 1 000 个用例集,每个用例集包含 300~1000 个用例.粗略地估计一下,大概要运行 50 多万个用例集,包含 3.2 亿个测试用例.如此庞大的数据量,如果每秒运行 10 个用例,也需要 8 800 小时以上,这是超过 1 年的持续运行时间.但是可以观察到,每个用例集中的用例都是 universe 中一些用例的组合.这样,只要先收集 universe 中所有用例的频谱并保存,对每个用例集,从 universe 中抽取出对应的频谱即可,只需要一个从用例映像查找频谱的索引.如此,可以大幅度地缩短实验时间.但要注意,可以这样处理的前提是:程序与时间无关并且未用到随机数生成,西门子程序集中的 7 个程序均符合这样的条件.

实验的具体流程如下:

(1) 对每个版本的程序,运行 universe 中的用例,收集程序频谱并加以储存;

(2) 对每种覆盖率类型,运行所有用例集.对每个用例集中的用例,根据索引找到其频谱;

(3) 在收集完用例集频谱之后,执行各种可疑度排序算法,并记录错误语句在各种算法下所排列的名次以及名次所占总代码数的百分比;

© 中国科学院软件研究所 http://www.jos.org.cn

(4) 所有测试结束后,根据记录的数据,统计各种算法的性能.

实际的实验中,还需要考虑对缺陷位置的标记.对不同可疑度排序方法进行比较,需要知道缺陷语句在其中的排名,这涉及到对缺陷位置的标记问题.由于各种算法所具有的不同特性,对可疑语句选取的倾向有所不同.西门子中引入缺陷的方式,一般是针对一条语句进行修改或者删除.如果将缺陷位置确切地定义在这些改变的位置,则在实验中会引发一些问题:对于缺失语句,其缺陷的位置无法定义;对于一些头文件中的宏定义错误,无法标记其位置,因为改动语句是不可执行的.对于一些变量声明错误(如 tot_info v10),某些方法倾向于将其后使用这个变量的分支排在前面,实际使用中,当检测语句遇到这样的分支时,实际上已经可以发现错误的语句.因此,仅仅将语句标记在代码中改动的位置上,可能是不太合理的.

图 1 是西门子中的两个代码片段(其中,第 10 行对 OLEV 的定义错误,由于它是不可执行的语句,所以将缺陷定义在第 118 行使用这个宏定义的位置;第 301 行对变量 *N* 的类型定义错误,在第 349 行和第 372 行对其有使用).在 tcas 程序错误版本 v13 中,第 10 行的宏定义引入了错误,但这条语句不可执行,没有频谱信息,因此将缺陷定义在第 118 行使用这个宏定义的地方.在 tot_info 程序错误版本 v10 中,第 301 行的变量类型定义错误,后面的代码,在第 349 行和第 372 行分别使用了这个变量.这两行在不同的基本块中,其频谱信息有所不同,不同的方法对它们排序的先后顺序也有差别.实际上,检查到其中任意一条,都可以发现变量定义的问题.因此,排序方法无论将哪条排在前面都是合理的.

tcas 的错误版本 v13

```
10  #define OLEV 600+100          /*bug*/
11  #define MAXALTDIFF 600        /*max altitude difference in feet*/
12  #define MINSEP 300            /* min separation in feet*/
13  #define NOZCROSS 100          /* in feet*/
    …

118 enabled=High_Confidence && (Own_Tracked_Alt_Rate<=
    OLEV) && (Cur_Vertical_Sep > MAXALTDIFF);
119 tcas_equipped=Other_Capability==TCAS_TA;
120 intent_not_known=Two_of_Three_Reports_Valid &&
    Other_RAC==NO_INTENT;
```

tot_info 的错误版本 v10

```
299 int i;                 /*row index*/
300 int j;                 /*column index*/
301 float N;               /*BUG (double)n*/
302 double info;           /*accumulates information measure*/
    …

349 N+=xi[i]=sum;
    …
372 info=N×log(N);        /*part 1*/
373
374 for (i=0; i<r; ++i)
```

Fig.1 Marks of fault position

图 1 缺陷位置的标记

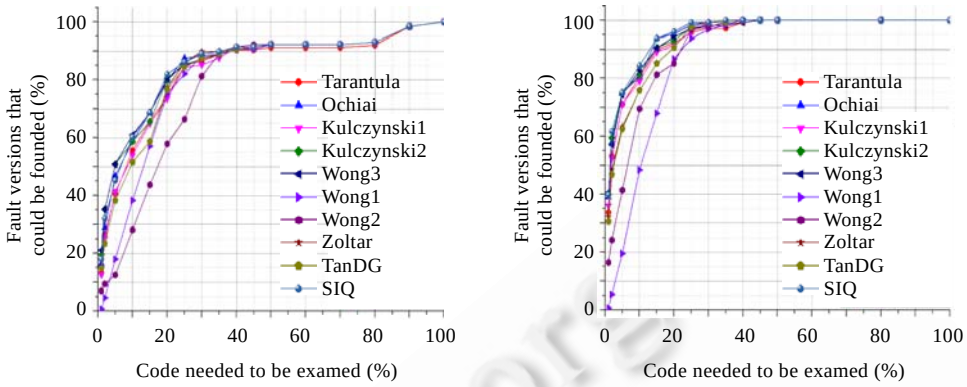
在实际程序中,这样的情况还有很多.为了更充分地反映不同方法的效果,本文对这类情况引入一种多缺陷位置标记的策略,即将检查语句可以直接找到缺陷的语句都进行标记,排序中,取这些标记语句中最靠前的作为可疑度排序方法的成绩.多缺陷位置标记中的语句是通过数据流定义的,设程序修改引入错误的点为 *S*,程序数据流图上,*S* 的直接后继和 *S* 本身作为多标记的对象.对应地,只标记引发错误所修改的语句,称为单缺陷位置定义策略.

多缺陷位置标记兼顾了缺陷定位的效果与各种方法的不同侧重.如图 1 所示,在西门子程序集中,有很多的缺陷版本应该使用多标记.相对于单标记,多标记更加全面.后面的实验中,会在两种标记策略下对各种方法进行比较.

© 中国科学院软件研究所 http://www.jos.org.cn

3.3 实验结果

对缺陷位置分别采用单缺陷位置标记和多缺陷位置标记的方法进行实验,在 universe 上进行,运行结果如图 2 所示.横坐标表示需要检查的代码行数占总行数的百分比,纵坐标表示发现的缺陷版本数占总版本数的百分比.对于缺陷定位,检查更少的代码、发现更多的错误是我们所预期的.



(a) 单缺陷位置标记 (b) 多缺陷位置标记

Fig.2 The resault of fault localization

图 2 各种方法的缺陷定位效果

在单缺陷位置标记的实验中,Wong 方法在检查代码数小于 5%时,有比较明显的优势;当检查代码数超过 10%时,SIQ 方法的效果开始超过 Wong 方法;当检查代码达到 20%时,SIQ 方法的效果超过 Wong 方法;当检查代码数达到 50%时,各种方法的曲线开始接近.在单缺陷位置标记下,Wong 方法在检查代码量很小(<5%)时有较好的表现;随着检查代码数的增加,SIQ 方法的效果优于 Wong 方法.

在多缺陷位置标记情况下,各种方法的效果比单缺陷位置标记有了一定程度的提升.SIQ 方法从一开始就一直有良好的效果,并始终领先于 Wong 方法和 Tarantula 方法的检查代码数;达到 50%以后,3 种方法都可以找到所有的缺陷,曲线重合.

在单标记和多标记的情况下,SIQ 方法都有良好的表现;特别是在多标记情况下,SIQ 方法的缺陷定位效果比其他方法有较为明显的提高.

西门子程序集提供了 7 种覆盖率类型,不同的覆盖率类型对缺陷定位的效果有不同的影响.同时,我们还在 space 程序上面进行了实验,由于 space 程序提供的用例集覆盖率类型与西门子不同,没有办法一一对应,所以只采用 space 的 cov 用例集进行实验.采用多缺陷位置标记的方法进行了实验,具体数据见表 4.

Table 4 Average performance

表 4 方法平均性能比较

	SIQ	Wong	Tarantula	Ochiai	Kulczyn1	Kulczyn2	Wong1	Wong2	Zoltar	TanDG
cov	***6.732***	7.449	6.933	6.798	8.099	6.772	13.038	8.246	6.769	6.778
rand	***6.004***	6.294	6.232	6.114	7.685	6.075	12.950	7.170	6.071	6.032
rand-covsize	***6.392***	6.685	6.505	6.420	8.128	6.400	13.158	7.091	6.398	6.394
bigcov	4.880	4.707	5.462	4.952	5.820	***4.702***	11.626	8.020	4.767	4.724
bigrand	4.918	5.141	5.572	5.076	5.973	***4.842***	11.803	8.081	5.005	4.957
universe	***4.177***	4.324	5.300	4.496	5.501	4.614	12.122	8.325	6.069	6.139
space	***8.858***	12.034	10.154	9.116	18.058	9.534	18.803	26.674	9.470	—

注:数字表示为了找到错误语句需要检查的代码数占总代码数的百分数,越小越好.

比较各种方法的平均性能,SIQ 方法除了在 bigcov 和 bigrand 覆盖率下效果稍逊于 Kulczynski2 方法,其他覆盖率下效果都是最好的.

© 中国科学院软件研究所 http://www.jos.org.cn

同时,对于不同的覆盖率类型,同一种方法的表现有所不同.大部分方法在 universe 情况下的效果最好,因为 universe 中包含了全部用例;各种方法在 bigcov 和 bigrand 下的表现明显比 cov,rand 和 rand-covsize 情况下要好.可以看出,用例数量对排序算法有较大的影响.同时,对于用例数量相近的覆盖率类型,排序方法的表现比较接近,rand- covsize 中每个用例集中用例的数量与 cov 中相应的用例集相同,只是用例随机选取.可以看到,虽然 rand-covsize 中用例无法保证一定的覆盖率,但其效果并不比 cov 差太多.rand 的表现甚至比 cov 还要好,因为 rand 中用例的数量比 cov 要多一些.cov 达到了分支覆盖,而 rand 不一定满足分支覆盖,但是 rand 的表现仍然好于 cov,这进一步说明了用例数量对缺陷定位效果的影响是显著的.同时也可以推测,缺陷定位效果与覆盖率可能是弱相关的,但这需要做进一步的实验来加以证明.

为了比较不同方法的稳定性,我们计算各种方法在所有版本上定位效果的标准差(见表 5).可以看到,在大部分情况下,SIQ 方法定位结果的标准差都是最小的,仅在 rand 和 rand-covsize 下排在第 2 位和第 3 位,总体的稳定性在实验的各种方法中都是最好的.

实验在西门子程序集和 space 程序上比较了 SIQ,Wong 和 Tarantula 等 10 种方法的缺陷定位效果,在不同缺陷语句标记方式以及不同的用例覆盖率类型下进行了实验.SIQ 方法在不同的实验配置下都能得到准确的结果,在不同的数据集上有很好的稳定性.

Table 5 Stability

表 5 方法稳定性比较

	SIQ	Wong	Tarantula	Ochiai	Kulczyn1	Kulczyn2	Wong1	Wong2	Zoltar	TanDG
cov	***6.609***	6.981	6.911	6.859	7.013	6.824	8.681	8.161	6.820	6.826
rand	5.910	***5.878***	6.285	6.225	6.463	6.178	8.688	7.325	6.172	6.026
rand-covsize	6.309	***6.113***	6.474	6.431	6.747	6.398	8.713	7.166	6.395	6.302
bigcov	***6.166***	6.295	6.715	6.465	7.050	6.325	7.871	7.664	6.293	6.275
bigrand	***5.854***	6.100	6.543	6.288	6.807	6.190	7.849	7.713	6.131	6.031
universe	***6.040***	6.394	7.400	6.505	7.661	6.969	7.516	8.016	7.570	7.605
space	***18.536***	19.163	18.641	20.199	27.100	20.737	18.625	30.396	20.739	—

注:数字对应方法在某种类型的覆盖率下,所有版本定位结果的标准差,越小越好.

4 结束语

基于频谱的缺陷定位方法由于所利用的信息易于收集且准确性很好,在近年的缺陷定位研究中受到了广泛重视.然而,现有的基于频谱的方法没有充分利用频谱信息,同时也没有能够根据测试数据动态调节算法,因此在不同数据集上的表现还不够稳定,很难得到准确的结果.

本文提出了基于事件信息量的缺陷定位方法——SIQ.SIQ 方法利用测试中事件的信息量,结合语句的执行信息,动态调节算法,提高了缺陷定位的效果.在西门子程序集和 space 程序的实验中,SIQ 方法的稳定性与适应性得到了验证.在与几种已有方法的比较中,SIQ 方法的缺陷定位效果也是最好的.

本文用 SIQ 方法的思想解释了 Tarantula 等方法.在后续工作中,我们将用 SIQ 方法的思想解释更多的基于频谱的缺陷定位方法,希望最终能为所有基于频谱的方法提供统一的理论框架.本文的实验使用的程序都是单错误版本,但实际的开发过程中更多的情况是多错误的.SIQ 方法在多错误程序上的缺陷定位效果还需要通过进一步的实验来加以考察.

致谢 在此,我们向对本文工作给予支持和建议的同行,尤其是周毓明老师和讨论班上的同学表示感谢.

References:

[1] Weyuker EJ. The cost of data flow testing: An empirical study. IEEE Trans. on Software Engineering, 1990,16(2):121−128. [doi: 10.1109/32.44376]

[2] Harrold MJ, Soffa ML. Efficient computation of interprocedural definition-use chains. ACM Trans. on Programming Languages and Systems, 1994,16(2):175−204. [doi: 10.1145/174662.174663]

[3] Santelices R, Harrold MJ. Efficiently monitoring data-flow test coverage. In: Proc. of the 22nd IEEE/ACM Int'l Conf. on Automated Software Engineering. 2007. 343−352. [doi: 10.1145/1321631.1321682]

© 中国科学院软件研究所 http://www.jos.org.cn

[4] Agrawal H, Horgan JR, London S, Wong WE. Fault localization using execution slices and dataflow tests. In: Proc. of the Int'l Symp. on Software Reliability Engineering. 1995. 143−151. [doi: 10.1109/ISSRE.1995.497652]

[5] Jones JA, Harrold MJ. Empirical evaluation of the tarantula automatic fault-localization technique. In: Proc. of the 20th IEEE/ACM Int'l Conf. on Automated Software Engineering. 2005. 273−282. [doi: 10.1145/1101908.1101949]

[6] Wong WE, Qi Y, Zhao L, Cai K. Effective fault localization using code coverage. In: Proc. of the 31st Annual Int'l Computer Software and Applications Conf. (COMPSAC 2007), Vol.1. 2007. 449−456. [doi: 10.1109/COMPSAC.2007.109]

[7] Hao D, Pan Y, Zhang L, Zhao W, Mei H, Sun J. A similarity-aware approach to testing based fault localization. In: Proc. of the 20th Int'l Conf. on Automated Software Engineering. 2005. 291−294. [doi: 10.1145/1101908.1101953]

[8] Hao D, Zhang L, Pan Y, Mei H, Sun J. On similarity-awareness in testing-based fault localization. Automated Software Engineering, 2008,15(2):207−249. [doi: 10.1007/s10515-008-0025-9]

[9] Zhang Z, Jiang B, Chan WK, Tse TH, Wang X. Fault localization through evaluation sequences. Journal of Systems and Software, 2010,83(2):174−187. [doi: 10.1016/j.jss.2009.09.041]

[10] Tan DG, Chen L, Wang ZY, Ding H, Zhou YM, Xu BW. Spectra-Based fault localization by increasing marginal weight. Chinese Journal of Computers, 2010,33(12):2335−2342 (in Chinese with English abstract)

[11] Naish L, Lee HJ, Ramamohanarao K. A model for spectra-based software diagnosis. ACM Trans. on Software Engineering and Methodology, 2011,20(3):Article 11. [doi: 10.1145/2000791.2000795]

[12] Abreu R, Zoeteweij P, van Gemund A. An evaluation of similarity coefficients for software fault localization. In: Proc. of the 12th Pacific Rim Int'l Symp. on Dependable Computing. 2006. 39−46. [doi: 10.1109/PRDC.2006.18]

[13] Abreu R, Zoeteweij P, van Gemund A. On the accuracy of spectrum-based fault localization. In: Proc. of the Testing: Academic and Industrial Conf. Practice and Research Techniques-Mutation (TAICPART-Mutation 2007). 2007. 89−98. [doi: 10.1109/TAIC.PART.2007.13]

[14] Gonzalez A. Automatic error detection techniques based on dynamic invariants [MS. Thesis]. Delft University of Technology, 2007.

[15] Lourenço F, Lobo V, Bação F. Binary-Based similarity measures for categorical data and their application in self-organizing maps. In: Proc. of the JOCLAD. 2004. http://www.isegi.unl.pt/docentes/vlobo/Publicacoes/publicacoes.htm

[16] Do H, Elbaum SG, Rothermel G. Supporting controlled experimentation with testing techniques: An infrastructure and its potential impact. Empirical Software Engineering, 2005,10(4):405−435. [doi: 10.1007/s10664-005-3861-2]

附中文参考文献:

[10] 谭德贵,陈林,王子元,丁晖,周毓明,徐宝文.通过增大边际权重提高基于频谱的错误定位效率.计算机学报,2010,33(12):2335−2342.

丁晖(1985－),男,辽宁鞍山人,博士生,主要研究领域为缺陷定位,程序分析.
E-mail: dinghui85@gmail.com

许蕾(1978－),女,博士,副教授,CCF 会员,主要研究领域为 Web 服务分析测试.
E-mail: xlei@nju.edu.cn

陈林(1979－),男,博士,讲师,CCF 会员,主要研究领域为软件分析维护.
E-mail: lchen@nju.edu.cn

徐宝文(1961－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为程序设计语言,软件工程,并行与网络软件.
E-mail: bwxu@nju.edu.cn

钱巨(1981－),男,博士,副教授,CCF 会员,主要研究领域为程序分析.
E-mail: jqian@nuaa.edu.cn

© 中国科学院软件研究所 http://www.jos.org.cn