Journal of Southeast University (English Edition) Vol. 30, No. 3, pp. 285 - 288 Sept. 2014 ISSN 1003—7985

Improved metrics for evaluating fault detection efficiency of test suite

Wang Ziyuan[1,2] Chen Lin[2] Wang Peng[3] Zhang Xueling[1]

([1] School of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210006, China)
([2] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)
([3] School of Computer Science and Engineering, Southeast University, Nanjing 210096, China)

Abstract: By analyzing the average percent of faults detected (APFD) metric and its variant versions, which are widely utilized as metrics to evaluate the fault detection efficiency of the test suite, this paper points out some limitations of the APFD series metrics. These limitations include APFD series metrics having inaccurate physical explanations and being unable to precisely describe the process of fault detection. To avoid the limitations of existing metrics, this paper proposes two improved metrics for evaluating fault detection efficiency of a test suite, including relative-APFD and relative-$APFD_C$. The proposed metrics refer to both the speed of fault detection and the constraint of the testing source. The case study shows that the two proposed metrics can provide much more precise descriptions of the fault detection process and the fault detection efficiency of the test suite.

Key words: software testing; test case prioritization; fault detection efficiency; metric

doi: 10.3969/j.issn.1003-7985.2014.03.005

The test case prioritization technique schedules test cases in an initial test suite in order, forming a prioritized test suite that increases its efficiency. Giving an existing initial test suite T_{init}, the test case prioritization technique aims to discover the best prioritized test suite $\sigma \in P$ such that

$$(\forall \sigma)(\sigma' \in P)(\sigma = \sigma')[f(\sigma) > f(\sigma')]$$

where P is the set of all the possible permutations of T_{init}, and f is an objective function[1].

An objective function called the average percent of faults detected (APFD) is usually utilized as the metric to evaluate the faults detection efficiency of the prioritized test suite $\sigma \in P$[1]. There are also some variants of the APFD metric, including NAPFD[2], $APFD_C$[3] etc. In this paper, we jointly call these metrics the APFD series.

APFD series metrics are designed for the test case prioritization problem, which implies the assumption $|\sigma| = |T_{init}|$ for each $\sigma \in P$. However, there may be some other types of scenarios, including the test case re-generation prioritization[4], time-aware test case prioritization[5], test case reduction[6], test goal prioritization[7] etc. In these above-mentioned scenarios, prioritized test suites under evaluation may contain only partial test cases in the given initial test suite, or sometimes may not be concerned with the initial test suite at all. APFD and its variants can hardly handle these situations.

For these problems, we propose an improved metric relative-APFD, which is related to a given testing resource constraint that determines how many test cases can be run, to replace the existing APFD and NAPFD. Furthermore, we also discuss the scenarios where test costs and fault severities are taken into consideration, and propose relative-$APFD_C$ to replace existing $APFD_C$. The case study shows that all the proposed metrics can provide much more precise illustrations of the fault detection efficiency of a prioritized test suite.

1 APFD Series Metrics

Let σ, Φ, and $TF(\phi, \sigma)$ be the prioritized test suite under evaluation, the set of faults contained in the software, and the index of the first test case in σ that exposes fault $\phi \in \Phi$, respectively, and then the APFD of σ is defined as[1]

$$APFD(\sigma) = 1 - \frac{\sum_{\phi \in \Phi} TF(\phi, \sigma)}{|\sigma||\phi|} + \frac{1}{2|\sigma|}$$

Sometimes, there may be non-detected faults that can not be detected by any test cases in σ. For each non-detected fault ϕ, Walcott et al.[5] set $TF(\phi, \sigma) = |\sigma| + 1$ as a penalty that may cause the APFD value to become negative. Cohen et al.[2] set $TF(\phi, \sigma) = 0$ for each non-detected fault, and proposed an improved metric normalized APFD (NAPFD) as

$$NAPFD(\sigma) = 1 - \frac{\sum_{\phi \in \Phi} TF(\phi, \sigma)}{|\sigma||\phi|} + \frac{p}{2|\sigma|}$$

Received 2013-12-28.

Biography: Wang Ziyuan (1982—), male, graduate, associate professor, wangziyuan@njupt.edu.cn.

Foundation items: The National Natural Science Foundation of China (No. 61300054), the Natural Science Foundation of Jiangsu Province (No. BK2011190, BK20130879), the Natural Science Foundation of Higher Education Institutions of Jiangsu Province (No. 13KJB520018), the Science Foundation of Nanjing University of Posts & Telecommunications (No. NY212023).

Citation: Wang Ziyuan, Chen Lin, Wang Peng, et al. Improved metrics for evaluating fault detection efficiency of test suite[J]. Journal of Southeast University (English Edition), 2014, 30(3): 285 - 288. [doi: 10.3969/j.issn.1003-7985.2014.03.005]

where p is the rate of faults detected by σ, i. e.,

$$p = \frac{|\{\phi \in \Phi \mid \mathrm{TF}(\phi, \sigma) \neq 0\}|}{|\Phi|}$$

Another improvement is to take the test costs and the fault severities into consideration. Let C_i be the cost of the i-th test case ($i = 1, 2, \ldots, |\sigma|$), and let S_ϕ be the severity of fault ϕ, and then the cost-cognizant weighted APFD ($\mathrm{APFD_C}$) is[3]

$$\mathrm{APFD_C}(\sigma) = \frac{\sum_{\phi \in \Phi}\left(S_\phi \sum_{i=\mathrm{TF}(\phi,\sigma)}^{|\sigma|} C_i - \frac{1}{2} C_{\mathrm{TF}(\phi,\sigma)}\right)}{\sum_{i=1}^{|\sigma|} C_i \sum_{\phi \in \Phi} S_\phi}$$

In recent years, people have proposed other metrics by extending APFD for special applications, including metrics for parallel processes[8], and metrics for evaluating the ratio of achieved efficiency[9] etc.

2 Limitations of APFD Series Metrics

2.1 Constraint on the sizes of test suites

We take test cases and faults in Tab. 1 as examples to show some incorrect results when using APFD series metrics in scenarios, where the sizes of prioritized test suites are varied.

Tab. 1 Faults detected by test cases

Test case	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8
T_1							×	×
T_2		×						
T_3		×		×			×	
T_4	×		×					
T_5				×	×	×		
T_6	×		×					×

1) For the situation where all faults are detected, we construct two prioritized test suites σ_1: T_3-T_5-T_2-T_4-T_1 and σ_2: T_3-T_5-T_6. Note that both σ_1 and σ_2 can detect all faults. Then we obtain the APFD values (see Fig. 1).

$$\mathrm{APFD}(\sigma_1) = \mathrm{APFD_C}(\sigma_1) = 0.6$$
$$\mathrm{APFD}(\sigma_2)\,\mathrm{APFD_C}(\sigma_2) = 0.5$$

However, it is incorrect to say that σ_1 is more efficient than σ_2. After run 1 (or 2) test case(s), both σ_1 and σ_2 detect 3 (or 5) faults; after run 3 test cases, σ_2 detects all the 8 faults while σ_1 detects only 5. This means that σ_2 detects faults more rapidly than σ_1.

2) For the situation where there are non-detected faults, we construct two prioritized test suites σ_3: T_3-T_2-T_5 and σ_4: T_3-T_5. Note that σ_3 and σ_4 detect the same faults. Then, we obtain NAPFD values.

$$\mathrm{NAPFD}(\sigma_3) = 0.354\,2$$
$$\mathrm{NAPFD}(\sigma_4) = 0.343\,6$$

It is also incorrect to say that σ_3 is more efficient than σ_4. After running 1 test case, both σ_3 and σ_4 detect 3 faults; after running 2 test cases, σ_4 detects 5 faults while σ_3 detects only 3. It means that σ_4 detects faults more rapidly than σ_3.

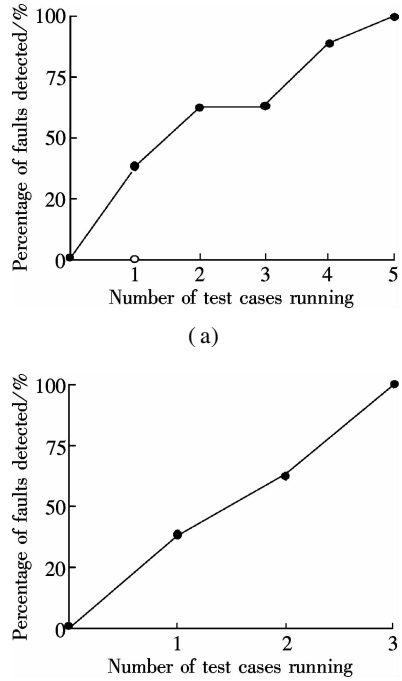


Fig. 1 Illustration of APFD. (a) $\mathrm{APFD}(\sigma_1)$; (b) $\mathrm{APFD}(\sigma_2)$

This limitation, which has been often overlooked previously, sometimes may lead to incorrect and confused experimental results in the applications of APFD series metrics[2, 5].

2.2 Process of fault detection

Another limitation is that the APFD series metrics cannot precisely illustrate the process of fault detection in the real world. They assume that during the running of one test case, the number of the newly detected faults (for APFD and NAPFD) or the total severities of the newly detected faults (for $\mathrm{APFD_C}$) grow linearly with consumed time. Factually, however, if a test case is still running, it cannot detect any faults since we cannot check whether it has passed or failed.

3 Improved Metrics

3.1 Relative-APFD

When comparing two prioritized test suites that contain different numbers of test cases, a fair testing resource should be provided first. Here the testing resource, which can be described as the positive integer m, can be considered as a constraint. If $m < |\sigma|$, at most m test cases in prioritized test suite σ can run. If $m > |\sigma|$, all test cases will run before the exhausting of testing resource. By using the testing resource constraint, we propose a metric relative-APFD. Evidently, it does not only depend on the test suites under evaluation, but also depends on the given testing resource constraint.

Formally, let σ, Φ, $\mathrm{TF}(\phi, \sigma)$ be the prioritized test suite under evaluation, the set of faults contained in the software and the position of the first test case in σ that exposes fault ϕ, respectively. We specifically set $\mathrm{TF}(\phi, \sigma) = 0$ for non-detected faults. For a given testing resource constraint m, the relative-APFD of σ is defined as

$$\mathrm{RAPFD}(\sigma, m) = p(m) - \frac{\sum_{\phi \in \Phi} \mathrm{TF}'(\phi, \sigma, m)}{m \mid \Phi \mid}$$

where

$$\mathrm{TF}'(\phi, \sigma, m) = \begin{cases} \mathrm{TF}(\phi, \sigma) & m \geqslant \mathrm{TF}(\phi, \sigma) \\ 0 & m < \mathrm{TF}(\phi, \sigma) \end{cases}$$

In addition, $p(m)$ is the ratio of the number of faults detected by the first m test cases in σ to the number of faults in Φ; i. e. ,

$$p(m) = \frac{\mid \{\phi \in \Phi \mid \mathrm{TF}'(\phi, \sigma, m) \neq 0\} \mid}{\mid \Phi \mid}$$

3.2 Relative-APFD$_C$

By considering the test costs, the given testing resource constraint should be scaled by a positive real number m_C. Then we can propose the metric relative-APFD$_C$ by extending relative-APFD.

Formally, let σ, Φ, $\mathrm{TF}(\phi, \sigma)$ be the prioritized test suite under evaluation, the set of faults contained in the software, the position of the first test case in σ that exposes fault ϕ, respectively. We specifically set $\mathrm{TF}(\phi, \sigma) = 0$ for non-detected faults. Additionally, let C_i be the cost of the i-th test case ($i = 1, 2, \ldots, \mid \sigma \mid$), and let S_ϕ be the severity of the fault ϕ. For a given testing resource constraint m_C, the relative-APFD$_C$ of σ is defined as

$$\mathrm{RAPFD}_C(\sigma, m_C) = p(m_C) - \frac{\sum_{\phi \in \Phi} \left(S_\phi \sum_{i=1}^{\mathrm{TF}'(\phi, \sigma, m_C)} C_i \right)}{m_C \sum_{\phi \in \Phi} S_\phi}$$

where

$$\mathrm{TF}'(\phi, \sigma, m_C) = \begin{cases} \mathrm{TF}(\phi, \sigma) & m_C \geqslant \sum_{i=1}^{\mathrm{TF}(\phi, \sigma)} C_i \\ 0 & m_C < \sum_{i=1}^{\mathrm{TF}(\phi, \sigma)} C_i \end{cases}$$

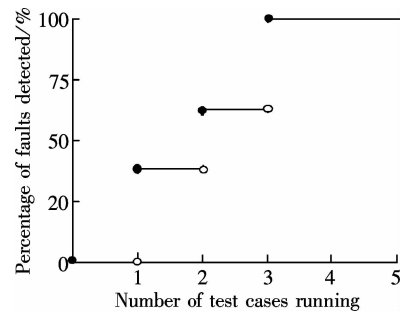
and $p(m_C)$ is the ratio of the total severities of faults detected by σ within the testing resource constraint to the total severities of all the faults in Φ; i. e. ,

$$p(m_C) = \frac{\sum_{\mathrm{TF}'(\phi, \sigma, m_C) \neq 0} S_\phi}{\sum_{\phi \in \Phi} S_\phi}$$

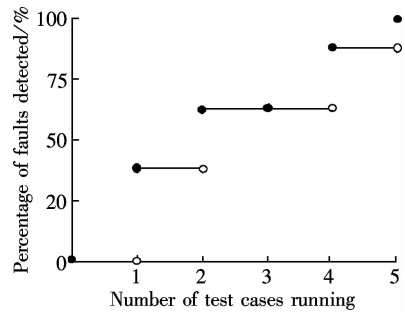
4 Case Study

Considering the prioritized test suites σ_1: T_3-T_5-T_2-T_4-T_1 and σ_2: T_3-T_5-T_6, their relative = APFD values for the testing resource constraint $m = 1, 2, 3, 4,$ and 5 are shown in Fig. 2 as the area under the step functions:

- $\mathrm{RAPFD}(\sigma_1, 1) = \mathrm{RAPFD}(\sigma_2, 1) = 0$;
- $\mathrm{RAPFD}(\sigma_1, 2) = \mathrm{RAPFD}(\sigma_2, 2) = 3/16$;
- $\mathrm{RAPFD}(\sigma_1, 3) = \mathrm{RAPFD}(\sigma_2, 3) = 1/3$;
- $\mathrm{RAPFD}(\sigma_1, 4) = 13/32 < \mathrm{RAPFD}(\sigma_2, 4) = 1/2$;
- $\mathrm{RAPFD}(\sigma_1, 5) = 1/2 < \mathrm{RAPFD}(\sigma_2, 5) = 3/5$.



(a)



(b)

Fig. 2 Illustrations of relative-APFD. (a) $\mathrm{RAPFD}(\sigma_1, m)$; (b) $\mathrm{RAPFD}(\sigma_2, m)$

The overall results show that, if the testing resource constraint is less than or equal to 3 (3 or less test cases run), σ_1 and σ_2 have the same efficiency; and if the constraint is greater than 3 (more than 3 test cases run), σ_2 is more efficient than σ_1.

Considering the other two prioritized test suites σ_3: T_3-T_2-T_5 and σ_4: T_3-T_5, their relative-APFD values for testing the resource constraint $m = 1, 2, 3$ are as follows:

- $\mathrm{RAPFD}(\sigma_3, 1) = \mathrm{RAPFD}(\sigma_4, 1) = 0$;
- $\mathrm{RAPFD}(\sigma_3, 2) = \mathrm{RAPFD}(\sigma_4, 2) = 3/10$;
- $\mathrm{RAPFD}(\sigma_3, 3) = 6/15 < \mathrm{RAPFD}(\sigma_4, 3) = 8/15$.

The overall results show that, if the testing resource constraint is less than or equal to 2 (2 or less test cases run), σ_3 and σ_4 have the same efficiency; if the constraint is greater than 2 (more than 2 test cases run), σ_4 is more efficient than σ_3.

The above two cases show that, relative-APFD avoids incorrect results obtained by existing APFD and NAPFD. The relative-APFD$_C$ has the same advantage, which is omitted here.

5 Conclusion

We make a brief review of widely used existing APFD

series metrics including APFD, NAPFD and $APFD_C$, and discuss their limitations. To avoid these, two improved metrics relative-APFD and relative-$APFD_C$ are proposed in this paper. These proposed metrics can illustrate the process of faults detection more precisely and practically, and provide more correct results to evaluate and compare the efficiency of prioritized test suites. In the future works, some metrics for a parallel testing process are required, since the cloud computing techniques have been widely applied to software testing.

References

[1] Rothermel G, Untch R H, Chu C Y, et al. Prioritizing test cases for regression testing [J]. *IEEE Transactions on Software Engineering*, 2001, **27**(10): 929 - 948.

[2] Qu X, Cohen M B, Woolf K M. Combinatorial interaction regression testing: a study of test case generation and prioritization [C]//*Proceedings of IEEE International Conference on Software Maintenance*. Paris, France, 2007: 255 - 264.

[3] Elbaum S, Malishevsky A G, Rothermel G. Incorporating varying test costs and fault severities into test case prioritization [C]//*Proceedings of the International Conference on Software Engineering*. Toronto, Canada, 2001: 329 - 338.

[4] Chen X, Gu Q, Zhang X, et al. Building prioritized pairwise interaction test suites with ant colony [C]//*Proceedings of the 9th International Conference on Quality Software*. Jeju, Korea, 2009: 347 - 352.

[5] Walcott K R, Soffa M L, Kapfhammer G M, et al. Time-aware test suite prioritization [C]//*Proceedings of 23rd International Symposium on Software Testing and Analysis*. Portland, Maine, USA, 2006: 1 - 11.

[6] Harrold M J, Gupta R, Soffa M L. A methodology for controlling the size of a test suite [J]. *ACM Transactions on Software Engineering and Methodology*, 1993, **2**(3): 270 - 285.

[7] Weiβleder S. Towards impact analysis of test goal prioritization on the efficient execution of automatically generated test suites based on state machines [C]//*Proceedings of the 11th International Conference On Quality Software*. Madrid, Spain, 2011: 150 - 155.

[8] Qu B, Xu B, Nie C, et al. A new metrics for test case prioritization in parallel scenario [J]. *Journal of Southeast University: Natural Science Edition*, 2009, **39**(6): 1104 - 1108. (in Chinese)

[9] Zhang X, Qu B. An improved metric for test case prioritization [C]//*Proceedings of Web Information Systems and Applications Conference*. Chongqing, China, 2011: 125 - 130.

改进的测试用例错误检测效率度量方法

王子元[1,2] 陈 林[2] 汪 鹏[3] 仉雪玲[1]

([1] 南京邮电大学计算机学院, 南京 210006)

([2] 南京大学软件新技术国家重点实验室, 南京 210093)

([3] 东南大学计算机科学与工程学院, 南京 210096)

摘要:分析了在测试用例优先级问题中被广泛用于度量测试用例集错误检测效率的 APFD 度量标准及其变种,指出 APFD 系列度量标准存在物理意义模糊、对错误检测过程描述不清晰等缺陷. 针对这些缺陷对已有度量标准进行改进,提出 2 种新的测试用例集错误检测效率度量方法 relative-APFD 和 relative-$APFD_C$. 新的度量方法在评价测试用例集效率时,综合考虑了错误检测速度和测试资源约束问题. 实例分析表明,新方法可以更为清晰地描述测试用例集错误检测过程,并更为准确地反映不同测试用例集的错误检测效率.

关键词:软件测试; 测试用例优先级; 错误检测效率; 度量方法

中图分类号:TP311