VAI TRÒ CỦA KỸ THUẬT ƯU TIÊN KIỂM THỬ
TRONG KIỂM THỬ PHẦN MỀM

Nguyễn Thanh Thảo - 5C20

GVHD: TS Vũ Minh Tuấn

**Tóm tắt:** Hiệu quả trong kiểm thử phần mềm đã trở thành một trong những điều được quan tâm nhiều hơn nhằm giảm thời gian thực hiện và chi phí của toàn bộ dự án. Bài viết về “Vai trò của kỹ thuật ưu tiên kiểm thử trong kiểm thử phần mềm” nhằm đưa ra tầm quan trọng của việc áp dụng các kỹ thuật ưu tiên trường hợp kiểm thử khác nhau trong việc thực hiện kiểm thử. Đồng thời, bài viết cũng nêu ra và nhấn mạnh những ưu điểm và nhược điểm cụ thể so với kỹ thuật kiểm thử không ưu tiên. Bằng cách tiến hành một nghiên cứu điển hình với sự kết hợp của cả phương pháp định lượng và định tính, bài viết đưa ra một số điểm sáng về vai trò của việc sử dụng các kỹ thuật ưu tiên trường hợp kiểm thử, nhằm giải quyết các câu hỏi nghiên cứu của bài.

**Từ khóa**: kiểm thử phần mềm; ưu tiên trường hợp thử nghiệm; nghiên cứu điển hình.

THE ROLE OF TEST CASE PRIORITIZATION
IN SOFTWARE TESTING

**Abstract:** Effectiveness in software testing has become one of the most essential points that is required to have more attention with the aim of lowering the execution time and expense of the entire project. The paper “The Role of Test Case Prioritization in Software Testing” shows the information about the importance of applying various test case prioritization techniques in executing test cases and emphases specific merits and demerits compared to non-prioritization testing. By conducting a case study with the corporation of both quantitative and qualitative methodologies, this paper figured out several key points for the role of utilizing test case prioritization techniques, which addressed its research questions.

**Keywords:** software testing; test case prioritization; case study.

1. Introduction

Software testing is an important aspect of the software development lifecycle, with the purpose of verifying and validating software to ensure that it meets requirements while maintaining quality. Effective testing is crucial for detecting shortcomings early, lowering post-release maintenance costs (Pareek & P. Kumar, 2020; A. Sawant et al., 2012). Test case prioritization is a method of prioritizing that involves grouping test cases based on a variety of factors, including priority, severity, and time constraints. Test case prioritization is important because it allows teams to boost testing productivity, save costs, and improve software quality (H. Do et al., 2010).

The study tries to resolve certain research questions about test case prioritization. There are 2 research questions including:

1. What is the importance of test case prioritization in software testing context?
2. What are the opportunities and potential risks associated with test case prioritization techniques?

The study intends to investigate the significance of test case prioritization on testing performance. Document analysis is utilized for collecting qualitative data, such as findings from test plans and reports. Quantitative analysis analyzes the influence of testing results using statistical metrics such as defect detection rate.

The paper is structured into three major sections: introduction, understanding and case study, and conclusion. The introduction provides context by outlining the study's statement, aims, scope, and methods. The understanding and case study section contains both a literature review and a case study. The conclusion chapters summarize the research findings and provide recommendations based on them.

2. Understanding and case study

2.1. Background and literature review

2.1.1. Fundamentals of software testing

Software testing, defined as the process of ensuring that software systems meet specifications, is a vital part of software development (D. Graham et al., 2018). It includes a number of tasks aimed at discovering and resolving problems in order to improve performance and fulfill customer expectations. Defects, which are typically human-caused, can range from minor to significant system failures, emphasizing the importance of comprehensive testing to minimize problems before software deployment.

There are several terminologies in the paper of “The Role of Test Case Prioritization in Software Testing”. It starts by defining a test case as a set of preconditions, test steps, and expected results derived from the software requirements definition. Errors are human activities that lead to incorrect system results, whereas defects are problems in system components that create unexpected events, which frequently arise during development as "bugs.".

2.1.2. Test case prioritization definition and techniques

Test Case Prioritization (TCP) is critical in software testing because it aims to find defects rapidly to save project costs. It involves arranging test cases based on certain criteria to improve testing effectiveness, reduce time and cost, and allocate resources efficiently.

TCP identifies important areas prone to errors, allowing testers to concentrate on high-risk portions of the system. TCP allows early identification of major defects by prioritizing relevant test cases, decreasing the overall financial investment and work required for bug rectification later in the development process. TCP, when combined with test case selection and reduction approaches, provides significant benefits throughout the development process. Understanding the significance of TCP is important for software testers in order to use it properly and gain great advantages in testing and software development.

There are several kinds of test case prioritization techniques including:

* Priority-based test case prioritizing: an essential way to determine the urgency of test cases. Priority is decided by factors such as software functionality, requirements, and tester experience.
* Risk-based test case prioritization: an important method in software testing for increasing system quality and efficiency. It involves evaluating potential risks based on probability and impact, then prioritizing them for testing.
* Requirement-based test case prioritization involves studying and discovering defects, inconsistencies, and potential risks in software requirements documentation. Testers must thoroughly comprehend the requirements, identifying the most important ones as a high priority.
* History-based test case prioritization: History-based approach is based on prior data on defect occurrence and resolution. It involves analyzing previous test results, bug reports, and resolution times to identify high-risk locations.
* Fault-proneness-based test case prioritization: Prioritize test cases based on potential problems in certain parts of software.

2.1.3. Previous research and case studies on test case prioritization

Ashima et al. (Ashima et al., 2020) examined several approaches to prioritizing test cases in order to improve software testing productivity. The findings show that automated prioritization enhances error identification, code coverage, and execution speed. Future research directions include researching larger datasets, enhancing existing algorithms, and investigating multi-objective strategies for test case selection. The document emphasizes the importance of test case optimization in cutting costs, increasing productivity, and improving fault detection rates in software testing.

J. Arafeen and H. Do's (2013) study examines the significance of requirements-based test case grouping in enhancing priority. The goals involve investigating if arranging by requirements enhances prioritization and comparing methods with controls. The results suggest that prioritizing test cases based on requirements criteria has considerable benefits for software companies, possibly improving overall testing efficiency and effectiveness.

V. Siqueira (2022) determines the efficiency of history-based prioritization approaches in regression testing. The idea is to evaluate how well these strategies work in this situation. The results suggest that the proposed methods are just as effective as ideal prioritization algorithms. Based on these findings, recommendations are offered for additional study into different regression test methods, running experiments to evaluate their usefulness, and comparing them to new choices.

R. Kavitha and N. Sureshkumar (R. Kavitha & N. Sureshkumar, 2010) present a test case prioritization technique for enhancing defect detection in regression testing. The primary goals include prioritizing test cases based on factors such as fault detection rate, with a final objective of increasing fault detection efficiency and overall software quality. Experimental results showed that the proposed technique enhanced random test case ordering, raising fault detection rates while reducing the number of test cases needed for problem identification. These findings support the use of the prioritizing method in enhancing defect identification within software testing processes, demonstrating its potential to improve software system consistency and robustness.

The study by C. Henard et al. (C. Henard et al., 2016) provides a thorough assessment of a number of test case prioritization strategies, including both white-box and black-box methods. It shows that white-box techniques perform marginally faster than black-box methods. In particular, all of the approaches completed the prioritization process in minutes, demonstrating their efficiency. The study recommends greater experimentation with other programs and failure scenarios to improve the findings' generalizability. Researchers might gain a better knowledge of the effectiveness and usability of various prioritization strategies in different settings by researching a wider range of software systems and failure kinds.

2.2. Case study: OpenCart system

2.2.1. Overview about case study

The case study investigates the effect of test case prioritization in the context of OpenCart, a popular e-commerce web platform focusing on technology products.

OpenCart was selected as the topic of this case study because it satisfied the case study's objectives. As an open-source e-commerce platform, OpenCart provides a wide range of features and capabilities for creating and customizing online stores. The case study's goal is to investigate the role of test case prioritization strategies within the larger context of software testing approaches. The study also aims to address particular research problems raised in the first chapters.

2.2.2. Implementation of test case prioritization in case study

In this section, it focuses on the implementation of test case prioritization within the OpenCart system case study. It discusses the implementation and adaptation of these techniques in the testing process, as well as the results obtained by using them in each case.

* OpenCart system is installed and tested locally following the installation on the OpenCart official website in this [LINK](https://docs.opencart.com/en-gb/installation/).
* The functional documentation and other requirements document are attached from the OpenCart official website in this [LINK](https://docs.opencart.com/en-gb/store-front/).
* Test plan of OpenCart system is attached to this document by clicking in this [LINK](https://drive.google.com/file/d/1o3jqqrqKmTNGgASey7Q49BaepNzxIN6M/view?usp=sharing).
* Test scenarios and test cases of OpenCart system are attached to this document by clicking in this [LINK](https://docs.google.com/spreadsheets/d/1tj0c8hozovd3FDXuk60sE0j_eKqHViza/edit?usp=drive_link&ouid=106436664421781642694&rtpof=true&sd=true).

From analyzing test scenarios and test cases documents, there are the list of test scenarios after apply prioritizing test case techniques:

* The Registration function: it is related to other system functionalities like Checkout, is prioritized due to its critical necessity.
* Checkout function: it is prioritized since it plays an important role in user transactions and business goals.
* Product page and category navigation function: it has their importance on user engagement and product discovery.
* Shopping cart functions: they are prioritized to reduce the risk of cart abandonment, with an emphasis on seamless user experiences and precise pricing.
* The homepage: it is the primary point of entry for users, is prioritized due to its importance in producing positive first impressions and easing navigation.
* Other capabilities, such as account settings and order history, are considered as well.

The author plans to measure APFD for each group to quantitatively evaluate the effectiveness of test case prioritization. The results will be presented in tables for visual analysis, facilitating comparisons between prioritized and non-prioritized testing. To illustrate, an example group of eight test cases is considered, with defects identified during regression testing and their positions recorded for APFD calculation.

The APFD formula:

$$1-(\frac{Tf\_{1}+Tf\_{2}+…+Tf\_{m}}{mn})+(\frac{1}{2n})$$

In which:

* $m$ is the number of faults in one test suite (test group)
* $n$ is the number of test case in one test suite
* $Tf\_{i}$ is the index position of the first test that the fault is uncovered.

We have 1 group of 8 test cases which is arranged and divided into 2 cases: prioritized test case list and non-prioritized test case list. They are both measured the APFD and illustrated as below:

Image. 1.1. Non-prioritized Test Case List

Image 1.2. Prioritized Test Case List

Respectively, the APFD of each case below is measured.

For Non-prioritized Test Case List:

$$1-\frac{3+2+2+0+2+8+0+2}{8×8}+\frac{1}{2×8} =0.7656$$

For Prioritized Test Case List:

$$1-\frac{1+6+1+0+5+3+0+1}{8×8}+\frac{1}{2×8} =0.7968$$

2.3. Findings

2.3.1. Analysis and interpretation of the results

The comparison of APFD values between the Non-prioritized Test Case List (0.7656) and the Prioritized Test Case List (0.7968) provides important information about the efficiency of test case prioritizing approaches in the OpenCart case study. A higher APFD rating suggests more efficient and effective bug identification in the system.

The Prioritized Test Case List has a higher APFD value than the Non-prioritized Test Case List, which directly addresses the first research question on the role of test case prioritizing in software testing. It illustrates how test case prioritizing strategies improve the efficiency and efficacy of problem identification, contributing to overall software quality.

Furthermore, the higher APFD result for the Prioritized Test Case List indicates that defects are detected earlier and more thoroughly than the Non-prioritized Test Case List. Early detection of faults is critical in software development because it allows for rapid debugging and resolution, preventing possible issues from expanding into larger difficulties later in the development process.

Also, the difference in APFD values gives statistical evidence for the benefits of test case prioritizing. It verifies the idea that prioritizing test cases based on specific criteria can improve bug detection results. It is especially important in the context of the OpenCart case study.

In contrast, there are several drawbacks to test case prioritizing strategies. Implementing test case prioritization takes extra time and effort to analyze and prioritize test cases based on multiple criteria. It can result in higher testing costs, especially in large-scale software projects. Prioritization aims to focus testing efforts on critical areas, however there is a risk of missing faults in lower-priority test cases. Bugs that are not high priority may go undetected until later stages of testing, or even after the product has been released.

2.3.2. Recommend for future studies

For future research on test case prioritization, the author may recommend that researchers investigate and develop new test case prioritization strategies that make use of advanced algorithms, machine learning, or AI. These innovative strategies may improve efficiency in discovering critical defects in software systems. Future research could look into the economic consequences of test case prioritization, such as the impact on testing costs and resource allocation. Analyzing the benefits and drawbacks of prioritization approaches and resource allocation could help companies in selecting an enhanced testing strategy.

Methods for automating the test case prioritization process could improve testing workflows while reducing manual effort. Automation tools or frameworks might be developed to automatically rank test cases based on criteria, improving productivity and scalability. Future study should focus on the practical challenges and obstacles of implementing prioritization algorithms in real-world software development projects. Understanding the problems might help practitioners develop solutions for overcoming these barriers and boosting the implementation of prioritization practices.

3. Conclusion

The conclusion summarizes the findings and results of the case study, emphasizing the importance of test case prioritization techniques in software testing and resolving 2 research questions. Through the case study, the role of test case prioritization has been highlighted as it improves the efficiency of discovering bugs in the system. It is also pointed out that the test case prioritization has the advantage of enhancing bug detection rating and the disadvantage in time and expense constraints. The author also suggests for future studies about test case prioritization, which should apply more advanced prioritizing techniques for a better consequence.

**REFERENCES**

1. A. Sawant, P. Bari, P. Chawan. (2012). Software Testing Techniques and Strategies. *International Journal of Engineering Research and Applications (IJERA)*, 980-986.

2. Ashima, G. Shaheamlung, K. Rote. (2020). A comprehensive review for test case prioritization in Software Engineering. *2020 International Conference on Intelligent Engineering and Management*, 331-336.

3. C. Henard, M. Papadakis, M. Harman, Y. Jia, Y. Le Traon. (2016). Comparing White-box and Black-box Test Prioritization. *ACM 38th IEEE International Conference on Software Engineering*, 523-534.

4. D. Graham, E. Veenendaal, I. Evans, R. Black. (2018). *Foundations of Software Testing: ISTQB Certification.*

5. H. Do, S. Mirarab, L. Tahvildari, G. Rothermel. (2010). The Effects of Time Constraints on Test Case Prioritization: A Series of Controlled Experiments. *IEEE Transactions on Software Engineering, 36*(5), 593-617.

6. J. Arafeen, H. Do. (2013). Test Case Prioritization Using Requirements-Based Clustering. *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation*, 312-321.

7. Pareek, P. Kumar. (2020). Development of Software Testing Techniques for Early Fault Detection. *Journal of Advancement in Parallel Computing*, 1.

8. R. Kavitha & N. Sureshkumar. (2010). Test Case Prioritization for Regression Testing based on Severity of Fault. *(IJCSE) International Journal on Computer Science and Engineering, 02*(05), 1462-1466.

9. V. Siqueira. (2022). History-based Prioritization in the Context of Manual Testing:. 15-80.