VIETNAM NATIONAL UNIVERSITY, HANOI
VNU **UNIVERSITY OF ENGINEERINGAND TECHNOLOGY**

SOCIALIST REPUBLIC OF VIETNAM
Independence – Freedom – Happiness

INFORMATION ON DOCTORAL THESIS

1. Full name : **Tran Nguyen Huong** 2. Sex: Male

3. Date of birth: 03/02/1979 4. Place of birth: Thai Binh

5.Admission decision number: Decision No. 778/QD-CTSV dated August 21, 2017, issued by the Rector of University of Engineering and Technology.

6.Changes in academic process:Decision No. 830/QD-DT dated August 31, 2018, issued by the Rector of the University of Engineering and Technology regarding the change of research topic for Ph.D. candidate Tran Nguyen Huong.

The old topic's title: "Specification and Verification of real-time Systems"

The new topic's title: "Improvements of the Automated Test Data Generation Method from Source Code"

7.Official thesis title: Improvements of the Automated Test Data Generation Method from Source Code

8. Major: Software Engineering 9. Code: 9480103.01

10.Supervisors: Assoc. Prof. Pham Ngoc Hung

11.Summary of the new findings of the thesis:

The thesis has achieved the following main results:

Firstly, the thesis improves the breadth-first search technique proposed in DART by combining Breadth-first search with static analysis. This improvement aims to improve the initial test data generation process by generating test data with high coverage and a reduced number of generated test data. When applying the breadth-first search technique, if the source code coverage does not increase after several iterations, the static analysis method is employed as a replacement. Furthermore, to reduce the test data generation time, a generalized test driver is proposed. This test driver performs compilation only once and could execute multiple test data sets. The test driver is expanded to handle various data types in C++ instead of being limited to C data types as in DART. The implemented tool is executed on several representative systems, demonstrating significant improvements in source code coverage, the number of solver invocations, meaningful test data sets, etc.

Secondly, the thesis proposes a method to generate test data using a Weighted Control Flow graph (WCFT). According to this method, after initializing the weights for the control flow graph, the execution path with the highest total weight would be selected to generate test data. If the constraints generated from the chosen path have a solution, then this path is marked as visited and the weights on its edges are increased by one. The process is repeated until all execution paths have been visited. This method has quick generation time and is capable of finding dead code. Next, to enhance the ability to generate test data at the boundary values, BVTG and IBVTG methods are proposed. These are two test data generation methods that are capable of detecting errors at the boundary points. With the BVTG method, the single conditions selected at the decision points of the control flow graph are normalized and fed to the SMT solver. This method generates test data that are capable of detecting errors at the boundary points. However, it is time consuming due to the use of a solver. The IBVTG method improves BVTG by directly generating test data at single conditions. Therefore, the generation time is greatly reduced. Finally, in order to have a method that can both generate test data to ensure coverage and detect errors at the boundary points, the thesis proposes the Hybrid method which is the combination of WCFT and IBVTG.

Thirdly, the thesis proposes a method of generating test data by emulating the source code unit. This is an improvement of the oriented test data generation method. This method is implemented in the control flow graph preprocessing step. Accordingly, each node that is a function call on the control flow graph will be replaced by a corresponding dummy variable. Execution paths generated from the control flow graph will be symbolically executed, constraint generation, and fed to a solver to generate test data. Experiments have shown that the generated test data have higher source code coverage, and the number of test datasets generated is less than the oriented test data generation method.

12. Practical applicability, if any:

The improvements in the thesiscould be applied to various programming languages. The thesis contributes theoretical solutions and supporting tools to enhance the effectiveness of automatic test data generation methods from source code for software projects, making automated testing methods more easily applicable in practice. The thesis contributes to improving the quality and reliability of software in general and within the research community in particular.

13. Further research directions, if any:

In the first study, the method should be improved with more features for C++ to be widely used in industrial projects. Firstly, generating test data for class templates and polymorphism is still considered a major challenge. The main reason is the difficulty in accurately detecting the called functions during execution. Secondly, the method would be expanded to generate test data for functions that contain exceptions. Specifically, the research will be extended to generate a series of test data that trigger runtime errors. Thirdly, the generalized C++ test driver will be improved to support various data types in C++, such as vectors, lists, etc. Lastly, symbolic execution needs to be enhanced to analyze statements using overloading mechanisms. Specifically, both the memory model and symbolic mapping should be strengthened to handle overloading, such as subtracting two instances of a class.

In the second study, the current Hybrid method only generates test data for primitive types. It needs to be extended to handle complex types such as pointers, structures, classes, etc. Determining the boundaries of these data types is also complex and is often presented in software specifications.

In the third study, it is necessary to continue improving the source code analysis technique and symbolic execution to reduce the test data generation time of AS4UT, especially in cases where functions have recursive calls or loops.

In each study within the thesis, the corresponding experimental tool is built and conducted. However, these tool research works are still separate, and the user interaction interface is limited. These tools should be integrated and upgraded in terms of the interface to create a unified tool that supports multiple test data generation methods. This tool will contribute to the wider application of proposed automated testing methods in practice and in industry.

14.Thesis-related publications:

[1].**T. N. Huong**, D. M. Kha, H.-V. Tran, and P. N. Hung. Generate Test Data from C/C++ Source Code Using Weighted CFG and Boundary Values. In 2020 12th Int. Conf. on Knowledge and Systems Engineering (KSE), pages 97–102, 2020.

[2]. **Tran Nguyen Huong**, Do Minh Kha, Hoang-Viet Tran and Pham Ngoc Hung. A Hybrid Method for Test Data Generation for Unit Testing of C/C++ Projects. VNU Journal of Science, Vol.39, No. 1 (2023).

[3]. D. A. Nguyen, **T. N. Huong**, H. D. Vo, and P. N. Hung. Improvements of Directed Automated Random Testing in Test Data Generation for C++ Projects. International Journal of Software Engineering and Knowledge Engineering, 29:1279–1312, 2019. (ISI Indexed)

[4]. **Tran Nguyen Huong**, Le Huu Chung, Lam Nguyen Tung, Hoang-Viet Tran, and Pham Ngoc Hung. An Automated Stub Method for Unit Testing C/C++ Projects. In 2022 14th Int. Conf. on Knowledge and Systems Engineering (KSE), pages 1-6 , 2022.