Canh Minh Do

JAIST Student Housing
1-8 Asahidai, Nomi, Ishikawa 923-1211 Japan
https://canhminhdo.github.io

Phone: (+84) 973-947-403
Email: canhdominh[at]jaist[dot]ac[dot]jp
Alt: minhcanh99[at]gmail[dot]com

PARTICULARS

EDUCATION

Japan Advanced Institute of Science and Technology — JAIST, Japan
A Ph.D. Candidate in Computer Science — *10/2019 - present*

Japan Advanced Institute of Science and Technology — JAIST, Japan
M.S. in Computer Science — *10/2018 - 09/2019*

VNU University of Engineering and Technology — VNU, Vietnam
M.S. in Computer Science — *10/2017 - 09/2018*

Aptech Computer Education — Aptech, Vietnam
Diploma in Information System Management (DISM) — *6/2011 - 12/2012*

National Economics University — NEU, Vietnam
B.Sc. in Computer Science — *10/2009 - 09/2013*

CURRENT STATUS

Japan Resident, Citizen of Vietnam.

RESEARCH INTERESTS

My research interests are mainly in the areas of formal methods, such as formal specification, interactive theorem proving and model checking, and tools supporting formal methods. Currently, I am focusing on mitigating the state space explosion in model checking and improving the running performance of model checking by parallelization. However, I am also open to conducting research on other interesting topics.

DISSERTATION

Title: "Parallelization of Formal Verification Tools"
Advisor: Professor Kazuhiro Ogata

Today, software systems are used in various applications where failure is unacceptable. Some subtle bugs lurking in such software systems may cause tragic loss of money, time, or even human life. Therefore, it is crucial to make such software systems highly trustworthy. To this end, formal verification is one promising way. Among some approaches to formal verification, model checking is the most successful one that is an automatic formal verification technique for verifying that a finite state system satisfies its desired properties by exploring the entire reachable state space and checking whether there are any violated states. However, there are still some challenges to tackle. One of them is the state space explosion problem, the most annoying one, which can make it impossible to conduct model checking experiments. Another challenge is to increase the running performance of model checking. The dissertation proposes some techniques to mitigate the state space explosion problem (space challenge) and improve the running performance of model checking (time challenge) to some extent by parallelization. We describe three non-trivial cases to demonstrate the proposed techniques: (1) parallelization of Java Pathfinder, a software model checker, for testing concurrent programs, (2) parallelization of Maude LTL model checker for checking leads-to properties, and (3) parallelization of Maude-NPA, a logical model checker, for cryptographic protocol analysis. We select three formal verification tools because of the following reasons: Java Pathfinder (JPF) is one of the most popular software model checkers for Java programs, which is developed by NASA; Maude LTL model checker is comparable to Spin in terms of both running time and memory consumption; and Maude-NPA is one formal verification/analysis tool dedicated to security protocol analysis that is very crucial for our current and future open network world, such as the Internet.

ACADEMIC HONORS

- Special Doctoral Research Fellow (DRF) Scholarship from JAIST for Ph.D. Program. 10/2019 - 09/2022.
- Scholarship From VNU-JAIST to Join the 1+1 Program from JAIST for Master Program. 10/2018 – 09/2019.
- Certificate of Hanoi Communist Youth Union for Top Graduating Students in Hanoi City. 09/2013.
- Highest GPA Graduating Student 2009-2013 in Computer Science Major at University. 09/2013.
- Third Prize in the Olympiad in Informatics of National Economic University. 10/2012.
- Fourth Prize in the Olympiad in Informatics of National Economic University. 10/2011.

WORK EXPERIENCE

- **Senior Software Engineer, TechAcademy Co ., Ltd**, Mar 2015 - Sep 2018. I have been involved in many projects that need to build both back-end and front-end parts. The former mainly uses CakePHP and Slim to develop APIs. The latter uses mostly AngularJS and ReactJS. Besides, using Selenium to build semi-automated testing for web applications. At the same time, I worked as a project manager for a team size of five people.

  Skills: PHP · CakePHP · Laravel · Slim · Javascript · AngularJS · ReactJS · NodeJS · Selenium · Automation Testing · Java · Python · Ruby · Ruby on Rails · Shell Scripting · Message Queue · MySQL · PostgreSQL · MongoDB · Git · Apache · Nginx · Linux.

- **Software Engineer, Dotoh Vietnam ., JSC**, Sep 2013 - Jul 2014. Building a GA-like analytic system that can detect which company user comes from and when visiting a website. It currently serves more than 100 high-traffic websites in Japan. Besides, joining to develop a digital document signing system that allows users to request a signature from others for documents such as contracts. Each user has to provide a valid certification on behalf of the user to sign documents online.

  Skills: PHP · Zend Framework · CakePHP · Javascript · Shell Scripting · Message Queue · MySQL · Git · Apache · Nginx · Linux.

- **Internship & Software Engineer, Langmaster ., JSC**, Nov 2012 - Jun 2013. Part of an initial team for developing an in-house CRM web application based on Zend framework that is used to facilitate the business operation at the company.

  Skills: PHP · Zend Framework · Javascript · MySQL · Apache.

RESEARCH EXPERIENCE

- **Parallel Specification-based Testing for Concurrent Programs**:

  Studies on testing concurrent programs have been conducted for nearly 40 years or even more. Compared to testing techniques for sequential programs, however, any testing techniques for concurrent programs do not seem mature enough. Moreover, many important software systems, such as operating systems, are in the form of concurrent programs. Therefore, testing techniques for concurrent programs must be worth studying so that they can be matured enough. We propose a specification-based testing for concurrent programs. For a formal specification S and a concurrent program P, state sequences are generated from P and checked to be accepted by S. We suppose that S is specified in Maude and P is implemented in Java. Java Pathfinder (JPF) and Maude are then used to generate state sequences from P and to check if such state sequences are accepted by S, respectively. Even without checking any property violations with JPF, JPF often encounters the notorious state space explosion while only generating state sequences. Thus, we propose a technique to generate state sequences from P and check if such state sequences are accepted by S in a stratified way. A tool is developed to support the proposed technique that can be processed naturally in parallel. Some experiments demonstrate that the proposed technique mitigates the state space explosion and improves the verification time, which cannot be achieved with the straightforward use of JPF.

  Skills: Java · JPF · RabbitMQ · Maude · Meta-programming · Redis · Master-worker Model.

- **A Parallel Version of a Support Tool for the Divide & Conquer Approach to Leads-to Model Checking**:

  Our research group has proposed the $L + 1$-layer divide & conquer approach to leads-to model checking ($L + 1$-DCA2L2MC), which is a new technique to mitigate the state space explosion in model checking. As shown by the name, $L + 1$-DCA2L2MC is dedicated to leads-to properties. We describe a parallel version of $L + 1$-DCA2L2MC and a tool that supports it. In a temporal logic called UNITY designed by Chandy and Misra, the leads-to temporal connective plays an important role and many case studies have been conducted in UNITY, demonstrating that many systems requirements can be expressed as leads-to properties. Hence,

it is worth dedicating to the properties. We also report on some experiments that demonstrate that the tool can increase the running performance of model checking. Counterexample generation is one of the main tasks in the tool that can be optimized to improve the running performance of the tool to some extent. We then propose a technique to generate all counterexamples at once that is based on the Tarjan algorithm, implemented in C++, and integrated into Maude, a language and system based on rewriting logic, so that users can use it easily. Some experiments are conducted to demonstrate the power of the technique that can improve the running performance of the tool. Furthermore, layer configuration selection affects the running performance of the tool. Therefore, we then propose an approach to finding a good layer configuration for the tool with an analysis tool that supports the approach. Some experiments are conducted to demonstrate the usefulness of the analysis tool as well as the approach for layer configuration selection.

Skills: C++ · Autotools · Maude · Maude Sockets · Meta-programming · Master-worker Model · Docker.

- **Parallel Maude-NPA for Cryptographic Protocol Analysis**

  With the emergence of the Internet and network-based services, many cryptographic protocols, also called security protocols, have been developed over decades to provide information security, such as confidentiality and authentication, in an insecure network. The design of cryptographic protocols, such as authentication protocols, is difficult, error-prone, and hard to detect bugs. Therefore, it is important to have automated tools to verify the properties of cryptographic protocols. Maude-NPA is a formal verification/analysis tool for analyzing cryptographic protocols in the Dolev-Yao strand space model modulo an equational theory defining the cryptographic operations, which starts from an attack state to find counterexamples or conclude that the attack concerned cannot be conducted by performing a backward narrowing reachability analysis. Although Maude-NPA is a powerful analyzer, its running performance can be improved by taking advantage of parallel and/or distributed computing when dealing with non-trivial protocols in which the state space is huge. We describe a parallel version of Maude-NPA in which the backward narrowing and the transition subsumption are parallelized at each layer. The tool supporting the parallel version has been implemented in Maude with a master-worker model. We report on some experiments of various kinds of protocols that demonstrate that the tool can increase the running performance of Maude-NPA by 44% on average for all non-trivial case studies in which the number of states located at each layer is considerably large.

  Skills: Maude-NPA · Maude · Maude Sockets · Maude Interpreters · Meta-programming · Master-worker Model · Docker.

- **A Learning Approach to Testing Concurrent Programs**

  Existing model checkers do not learn from the explorations performed in their previous work at all. Meanwhile, much progress has been advanced in machine learning technologies for a decade, especially deep learning. Given a concurrent program for testing, JPF is used to model check the program and generate traces up to a depth where traces are sequences of Java bytecode instructions. Word2Vec inspired us to build an encoding model called Instr2Vec for vector representations of Java bytecode instructions. Each trace will be encoded by Instr2Vec and fed into a machine learning model that formulates bug detection as a binary classification problem by using deep learning to train the classifier that can distinguish correct from incorrect traces. The trained classifier will score traces such that higher scores are given to traces that are more likely to lead to failures. We control JPF to first test the traces that start with states given higher scores, making it quicker to detect failures lurking in concurrent programs at deep positions. Unfortunately, this research direction has been postponed because it produced a poor result. We may restart it if we come up with a better way to encode testing problems for concurrent programs.

  Skills: Java · JPF · Python · TensorFlow · Deep Learning.

PUBLICATIONS

CONFERENCE PAPERS

1. Canh Minh Do and Kazuhiro Ogata, "Specification-based Testing with Simulation Relations", *The 31st International Conference on Software Engineering and Knowledge Engineering (SEKE)*, 2019.
2. Canh Minh Do and Kazuhiro Ogata, "A Divide & Conquer Approach to Testing Concurrent Java Programs with JPF and Maude", *9th International Workshop SOFL+MSVL*, 2019.
3. Yati Phyo, Canh Minh Do and Kazuhiro Ogata, "Toward development of a tool supporting a 2-layer divide & conquer approach to leads-to model checking". *International Conference on Advanced Information Technologies (ICAIT)*, 2019.
4. Canh Minh Do and Kazuhiro Ogata, "Parallel Stratified Random Testing for Concurrent Programs", *The 20th IEEE International Conference on Software Quality, Reliability, and Security (QRS)*, 2020.

5. Canh Minh Do and Kazuhiro Ogata, "A Divide & Conquer Approach to Testing Concurrent Programs with JPF", *The 27th Asia-Pacific Software Engineering Conference (APSEC)*, 2020.
6. Yati Phyo, Canh Minh Do and Kazuhiro Ogata, "A Support Tool for the L+1-Layer Divide & Conquer Approach to Lead-to Model Checking", *Intelligent and Resilient Computing for a Collaborative World, 45th Anniversary Conference (COMPSAC)*, 2021.
7. Yati Phyo, Canh Minh Do and Kazuhiro Ogata, "A Divide & Conquer Approach to Conditional Stable Model Checking", *18th International Colloquium on Theoretical Aspects of Computing (ICTAC)*, 2021.
8. Canh Minh Do, Yati Phyo, Adrián Riesco and Kazuhiro Ogata, "A Parallel Stratified Model Checking Technique/Tool for Leads-to Properties", *The 7th International Symposium on System and Software Reliability (ISSSR)*, 2021.
9. Canh Minh Do, Yati Phyo, Adrián Riesco and Kazuhiro Ogata, "Parallel Maude-NPA for Cryptographic Protocol Analysis", *14th International Workshop on Rewriting Logic and its Applications (WRLA)*, 2022 (to appear).
10. Canh Minh Do, Yati Phyo and Kazuhiro Ogata, "A Divide & Conquer Approach to Until and Until Stable Model Checking", *The 34th International Conference on Software Engineering & Knowledge Engineering (SEKE)*, 2022 (to appear).
11. Moe Nandi Aung, Yati Phyo, Canh Minh Do and Kazuhiro Ogata, "A Tool for Model Checking Eventual Model Checking in a Stratified Way", *The 9th International Conference on Dependable Systems and Their Applications (DSA)*, 2022 (to appear).

JOURNAL PAPERS

12. Yati Phyo, Canh Minh Do and Kazuhiro Ogata, "A Divide & Conquer Approach to Leads-to Model Checking", *The Computer Journal*, 2021.
13. Moe Nandi Aung, Yati Phyo, Canh Minh Do and Kazuhiro Ogata, "A Divide & Conquer Approach to Eventually Model Checking", *MDPI Journal Mathematics*, 2020.
14. Thet Wai Mon, Dang Duy Bui, Duong Dinh Tran, Canh Minh Do and Kazuhiro Ogata, "Graphical Animations of the NS(L)PK Authentication Protocols (Research Note)", *Journal of Visual Language and Computing*, 2021.
15. Canh Minh Do and Kazuhiro Ogata, "Parallel Specification-based Testing for Concurrent Programs", *IEEE Access*, 2022.

PAPERS UNDER REVIEW

16. Yati Phyo, Canh Minh Do and Kazuhiro Ogata, "A Tool for Model-checking Conditional Stable Properties in a Layered Way", *TBA*, 2021.

PENDING PAPERS

17. Canh Minh Do, Yati Phyo, Adrián Riesco and Kazuhiro Ogata, "A Parallel Stratified Model Checking Technique/Tool for Leads-to Properties (Extended Version)", *TBA*, 2022.
18. Canh Minh Do, Yati Phyo, Adrián Riesco and Kazuhiro Ogata, "Parallel Maude-NPA for Cryptographic Protocol Analysis (Extended Version)", *TBA*, 2022.
19. Canh Minh Do, Yati Phyo and Kazuhiro Ogata, "A Parallel Technique/Tool for Model Checking Conditional Stable Properties in a Layered Way", *TBA*, 2022.

APPLICATIONS

These applications were developed when I have been doing research at JAIST. They are all freely available for use in the public domain.

1. **Spec-based**
   - An environment for testing concurrent Java programs based on specifications by using JPF and Maude.
2. **DCA2MC**
   - $L + 1$ layer divide & conquer approach to model checking leads-to, eventual, conditional stable, until and until stable properties.

- The tool supports running in both sequential and parallel modes.

3. **Parallel-maude-npa**
   - A parallel version of Maude-NPA for cryptographic protocol analysis.
4. **SMGA-Plus**
   - A graphical animation tool for cryptographic protocols.
5. **CafeOBJ VSCode Extension**
   - Visual studio code extension for CafeOBJ language, includes syntax highlighting and snippets.

LANGUAGES

Proficient in English (TOEIC 865), basic Japanese (N5) and native Vietnamese.

REFERENCES

FROM ACADEMIA

Prof. Kazuhiro Ogata
Professor
School of Information Science
Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi, Ishikawa 923-1292 Japan
Phone: (+81) 761-51-1211
ogata[at]jaist[dot]ac[dot]jp

Prof. Adrián Riesco
Associate Professor
Department of Software Systems and Computation
Universidad Complutense de Madrid
Madrid, Spain
Phone: (+34) 91-394-7648
ariesco[at]fdi[dot]ucm[dot]es

FROM INDUSTRY

Mr. Kei Akatsuka
Chief Executive Officer
KEIT., Inc
Chiyoda-ku, Tokyo 101-0032 Japan
Phone: (+81) 362-40-9446
keit[dot]akatsuka[at]keit[dot]co[dot]jp

Mr. Nguyen Van Nam
Chief Technology Officer
Langmaster., JSC
485 Hoang Quoc Viet, Cau Giay, Ha Noi, Vietnam
Phone: (+84) 902-171-042
namnv[at]langmaster[dot]edu[dot]vn