PhD in Information Technology and Electrical Engineering
Università degli Studi di Napoli Federico II

PhD Student: Francesco Altiero

Cycle: XXXVI

Training and Research Activities Report

Year: First

Tutor: prof. Adriano Peron

Co-Tutor: prof. Anna Corazza

Date: October 28, 2021

Training and Research Activities Report

PhD in Information Technology and Electrical Engineering

Cycle: XXXVI **Author: Francesco Altiero**

1. Information:

- **PhD student:** Francesco Altiero
- **DR number:** 995043
- **Date of birth:** 07/07/1986
- **Master Science degree:** Computer Science **University:** Federico II
- **Doctoral Cycle:** XXXVI
- **Scholarship type:** *UNINA*
- **Tutor:** prof. Adriano Peron
- **Co-tutor:** prof. Anna Corazza

2. Study and training activities:

Activity	Type[1]	Hours	Credits	Dates	Organizer	Certificate[2]
Digital Forensics	Course	10	3	03-05-06-09-10.10.2021	Dr. Giovanni Cozzolino – DIETI UNINA	Y
Robot Manipulation and Control	Seminar	2.5	0.5	17.11.2020	Prof. Bruno Siciliano – DIETI, UNINA	Y
Picariello Lectures IV: #andratuttobene - Image, Texts, Emojis, & Geodata in a Sentiment Analysis Pipeline	Seminar	1.5	0.3	25.11.2020	Prof. Flora Amato - DIETI, UNINA	N
ITEE Webinar: Patent searching	Seminar	1	0.2	27.11.2020	Prof. Alessandra Scippa – DIETI, UNINA	Y
ITEE Webinar: How to get Published by IEEE	Seminar	1.5	0.3	02.12.2020	Prof. Alessandra Scippa - DIETI, UNINA	Y
Network Systems, Kuramoto Oscillators, and Synchronous Power Flow	Seminar	1.5	0.3	03.12.2020	Ph.D. Marco Coraggio - DIETI, UNINA	N
Picariello Lectures VI: Exploiting Deep Learning and Probabilistic Modeling for Behavioural Analytics	Seminar	1.5	0.3	09.12.2020	Prof. Flora Amato – DIETI, UNINA	Y

Picariello Lectures VII: Data Driven Transformation in WINDTRE through Manager's voice	Seminar	1.5	0.3	17.12.2020	Prof. Flora Amato – DIETI, UNINA	Y
GDPR Basis for Computer Scientists	Seminar	1.5	0.3	10.12.2020	Prof. P.A. Bonatti – DIETI, UNINA	Y
Picariello Lectures VIII: From Photometric Redshifts to Improved Weather Forecasts - an Interdisciplinary view on Machine Learning	Seminar	1	0.2	13.01.2021	Prof. Flora Amato – DIETI, UNINA	Y
Picariello Lectures X: AI LEGAL: Artificial Intelligence for notary's sector - A case study	Seminar	1	0.2	27.01.2021	Prof. Flora Amato – DIETI, UNINA	Y
Picariello Lectures XII: Machine Learning: causality lost in translation	Seminar	1.5	0.3	10.02.2021	Prof. Flora Amato – DIETI, UNINA	Y
Statistical Data Analysis for Science and Engineering Research	Course	14	4	17-19-24-25.03.2021 – 3-4.03	Prof. R. Pietrantuono - DIETI, UNINA	
Data Science for Patient Record Analyis	Course	8	2.5	10-17-24.02.2021 – 02-03-17.03.2021	Prof. M. Cinque – ITEE & ICTH, UNINA	Y
Robo Ludens: A game design taxonomy for human-robot interaction	Seminar	1	0.2	05.03.2021	Prof. Silvia Rossi - PRISCA Lab. DIETI, UNINA	N
Dai Mainframe all'IoT: una retrospettiva sull'evoluzione delle architetture di calcolo	Seminar	2	0.4	08.03.2021	Prof. A. Cilardo - DIETI, UNINA	Y
Scientific Programming and Data Visualization with Python	Course	8	2	08-10.03.2021	Prof. Alessio Botta – DIETI, UNINA	Y
Combinatorial Optimization	Course	40	6	03.2021 – 06.2021	Prof. Paola Festa – DMA, UNINA	Y
Data Analytics	Course	40	6	03.2021 – 06.2021	Prof. A. M. Tulino – DIETI, UNINA	Y
Picariello Lectures XVII: The coming Revolution of Data driven Discovery	Seminar	1.5	0.3	25.03.2021	Prof. Flora Amato – DIETI, UNINA	Y
Introduction to Underwater Robotics	Seminar	2	0.4	18.05.2021	DIETI, UNINA	Y
5G: l'architettura, le applicazioni e la rete di accesso radio	Seminar	2	0.4	08.06.2021	DIETI, UNINA	Y

L'esposizione ai campi elettromagnetici generati dal sistema 5G - Metodologie scalari e vettoriali di misura dell'esposizione e tecniche di estrapolazione	Seminar	4	0.8	16.07.2021	Prof. N. Pasquino – DIETI, UNINA	Y
Localized least-squares radial basis function methods for PDEs	Seminar	1	0.2	05.10.2021	Prof. Daniela di Serafino – DMA, UNINA	Y
Analyzing and Supporting the Evolution of Data-Intensive Systems	Seminar	1	0.2	07.10.2021	Prof. G. Bavota - Software Institute Università della Svizzera Italiana	N
Visualizing Discord Servers - definitely not a virtual conference video replay	Seminar	1	0.2	14.10.2021	Ph.D. Alejandro M. Rozo - Software Institute Università della Svizzera Italiana	N
Qiskit: state of the art and tools for Quantum Computers from IBM	Seminar	2	0.4	15.10.2021	Prof. A.S. Cacciapuoti – DIETI, UNINA	Y
SAR Polarimetry: Theory, Machine Learning & Applications	Seminar	2	0.4	19.10.2021	Prof. A. Iodice – DIETI, UNINA	Y

1) Courses, Seminar, Doctoral School, Research, Tutorship
2) Choose: Y or N

2.1. Study and training activities - credits earned

	Courses	**Seminars**	**Research**	**Tutorship**	**Total**
Bimonth 1	**0**	**2.2**	**7**	**0**	**9.2**
Bimonth 2	**3**	**0.7**	**6**	**0**	**9.7**
Bimonth 3	**8.5**	**0.7**	**3**	**0**	**12.2**
Bimonth 4	**0**	**0.8**	**8**	**0**	**8.8**
Bimonth 5	**0**	**0**	**6**	**0**	**6**
Bimonth 6	**12**	**2.7**	**3**	**0**	**14.7**
Total	**23.5**	**7.1**	**33**	**0**	**63.6**
Expected	**20-40**	**5-10**	**10-35**	**0 – 1.6**	

3. Research activity:

I am continuing the studies on the problem which I coped with in my master's degree thesis, namely the source code analysis and its applications in software testing. In particular, I am focusing on *Regression Test Prioritization* problem. Regression Test Prioritization concerns strategies and techniques which re-order the test-cases in a software's test-suite when a new version of the software is released. These techniques aim to give greater priority to test-cases which have greater probability to exploit regression faults, i.e., faults which are introduced in software parts which were validated in previous versions.

Strategies for Regression Test Prioritization are widely used solutions in cases when there are limited resources (e.g., time or money) to perform the verification phase of a new software version, often due to time-to-market constraints. In this case, tests are executed until the allocated resources for software testing are expired. Thus, the test-suite is not entirely executed, and it is of primary importance to execute as many tests which can detect defects as possible.

I am currently working on novel prioritization techniques considering the variations between the source code of two versions of the software, namely the *code-churn*. Initially, I performed an extensive study on the state-of-the-art literature on the topic, both to investigate employed techniques and to define the common methodology of research in the field. Although some Regression Test Prioritization strategies in literature use information on source code changes between versions, many of them consider only textual variations (e.g., edit-distance or *diffs*). I am working on techniques which consider also the "nature" of the code change, based on the intuition that some kind of variations are more likely prone to introduce faults. For example, a change in the condition of a branch construct can has a higher likelihood to introduce faults than the renaming of a variable. To this purpose, I employed *Tree Kernels* to design these prioritization techniques. Tree Kernels are a class of functions which compute similarity between tree models and have been widely used in Natural Language Processing due their capability to exploit both the lexical and structural similarity of sentences. Starting from the source code of two versions of a software, I modeled a pipeline to prioritize the last version's test-suite using these novel techniques. In the first place, both versions of the software are converted in their *Abstract Syntax Tree* representation, which is a tree structure that highlights the code construct structural dependency and their tokens. Subsequently, I employed several Tree Kernel functions to evaluate the similarity between tree representations of source code. The resulting similarity scores are merged with *per-test coverage report* to assign a rank to each test-case in the test-suite of the new version. A higher rank is assigned to test-cases which covers software parts which are less similar between the two versions. Test-cases are then sorted by descending order of their ranks. I am furthermore designing new types of Tree Kernels which are more suitable for source code analysis and defect prediction which can be employed in the prioritization pipeline. The work on Regression Test Prioritization by me and my research group led to the publication of the conference paper "*Inspecting code churn to prioritize test cases*", presented at the *32nd International Conference on Software and Systems* (ICTSS 2020).

After a comprehensive study of literature on the Regression Test Prioritization topic, I chose to follow an empirical methodology for my research, as this methodology is widely used in the field. To the purpose of quantitatively compare performances of Tree Kernel techniques with other state-of-art prioritization strategies, I chose the *Average Percentage of Fault Detected* (APFD) metric. APFD has been widely used in the RTP field to evaluate the fault-detection performance of a re-ordered test-suite produced by a RTP technique and measures the velocity of finding faults in a software by the prioritized test-suite. For this reason, the software projects used as subjects of the prioritization experiments should had faults in some modules along with information related to the fault (e.g., which test-case exploited which fault). To perform comparison between techniques, statistical analysis (e.g., Mann-Whitney U test) between APFD values of these technique have been applied.

After deciding a performance metric, I collected a dataset of several benchmark software projects which were used in different publications to compare the Tree Kernel prioritization techniques with state-of-art strategies. The collection process encountered different problems, as many datasets used in literature were not replicable. For example, software projects were often obsolete or lacked information needed in

for my prioritization techniques (e.g., coverage data). Thus, I selected different project from different datasets in literature to create a new dataset on which execute prioritization experiments. Furthermore, when faults were not available, I injected faults in the software by generating software mutants. In almost all cases, per-test coverage information needed to be computed, to know precisely which test covers which line of code. I analyzed several tools which evaluated code-coverage, but none of them was capable to produce per-test information. Thus, I extended *Open Clover*, an open-source software used to evaluate coverage, adding the possibility to produce per-test line coverage XML report.

With my research group, I've implemented the pipeline of Regression Test Prioritization I designed, producing the *Prioritization* platform. The platform was written in Java and works on software coded in the same language. The platform supports some baseline prioritization techniques along with the novel Tree Kernel techniques and automatizes all the prioritization steps. The platform has been used to evaluate the novel techniques against the baseline and the results have been significatively better than baseline techniques implemented. I am currently working on the implementation in the platform of other state-of-art prioritization techniques to compare Tree Kernel prioritization strategies with a wider range of strategies.

I am also collaborating with my research group on the extension of the benchmark dataset. To this purpose, a software-mining tool is currently in development. The tool is designed to automatically download software projects and their versions from common code-sharing or CI/CD platform, such as *GitHub* and *TravisCI*. This kind of extension allows the evaluation of the performances of prioritization techniques also on software which are developed with these methodologies.

Aside from my main research field, I am currently working on the application of *Machine Learning* techniques in the medical field, specifically to predict severity level of autism in patients, with a research team led by Prof. Marcello Cinque and including ITEE and ITCH students. We employed *Naïve Bayes* and *Decision Tree* models performed experiments on a small dataset of children diagnosed with autism, obtaining good measurement of accuracy by including also genetical alteration as a feature. As far as we know, no previous study in the literature aims to predict the severity level and employs genetical features. Furthermore, I am working with Prof. Sergio di Martino and Dr. Antonio Origlia on *software visualization* topics. Our idea is to use *Unreal Engine* to produce a 3D visualization of a project's codebase which shows changes between two subsequent *commits* and highlights variations which can be more fault prone. Using the *Unreal Engine* capabilities, we are currently investigating the possibility to add both animations and sounds as a further media to software presentation.

4. Research products:

Conference Paper: Altiero F., Corazza A., Di Martino S., Peron A., Starace L.L.L. "Inspecting Code Churns to Prioritize Test Cases". In: Casola V., De Benedictis A., Rak M. (eds) International Conference on Testing Software and Systems. ICTSS 2020, Naples. Lecture Notes in Computer Science, vol 12543. Springer, Cham; Published 2020.

Prototype Software: I designed and implemented *Prioritization*, a Java platform used to perform experiments on Regression Test Prioritization with different strategies, including the novel Tree Kernel Prioritization techniques currently main topic of my research.

Tool extension: I extended *Open Clover*, an open-source tool to perform code coverage and to obtain reports. The extension I made adds the capability to produce per-test coverage information with a line of code level granularity.

5. Conferences and seminars attended

Conference: *International Conference on Testing Software and Systems – ITCSS 2020*, Naples, 9-11.12.2020; presented the paper "Inspecting Code Churns to Prioritize Test Cases".

6. Activity abroad:

None. 0 months spent abroad.

7. Tutorship

None.