GILBERT PAJELA
Hunter College, City University of New York
695 Park Avenue, HN 1000G
New York, NY 10065
Email: gpajela AT gradcenter DOT cuny DOT edu

Education

2019 THE GRADUATE CENTER, CITY UNIVERSITY OF NEW YORK
Doctor of Philosophy in Computer Science, Expected May 2019

2013 NEW YORK INSTITUTE OF TECHNOLOGY
Master of Science in Computer Science, May 2013

1996 UNIVERSITY OF PENNSYLVANIA
Bachelor of Science in Electrical Engineering, December 1996

Research Interests

Formal Methods, Software Verification, Model Checking, Static Analysis, Program Analysis, Software Security, Data Mining, Big Data, Machine Learning, Cryptography, Artificial Intelligence

Publications

Shankar, S., & Pajela, G. (2016, April). A Tool Integrating Model Checking into a C Verification Toolset. In *International Symposium on Model Checking Software* (pp. 214-224). Springer International Publishing.

Work Experience

Fall 2016– Supervised Programming Lab, Hunter College, New York, NY

- *Instructor for two sections of a programming lab designed to teach students how to apply principles of design and analysis in creating substantial programs and to give students deep practical knowledge of C++ and the Linux operating system.*

2015– Research Assistant, Hunter College, New York, NY

- *Assisted with development of a plugin written in OCaml for Frama-C, an extensible and collaborative platform dedicated to source-code analysis of C software.*
- *Assisted with developing ways of integrating static analysis with model checking.*
- *Produced regular reports on research progress.*

Honors

2016 Attended SSFT16 (Sixth Summer School on Formal Techniques 2016): *A week-long program targeting graduate students and young researchers interested in developing and using formal techniques in their research.*

2013 –2014 Member, Upsilon Pi Epsilon (UPE): *International honor society for students in the Computer and Information Sciences*

Projects and Relevant Coursework

Fall 2016 Abstract Interpretation-Based Approaches to Security (Principles of Software Security Course Project)

- *Presented a description and summary on the framework of Abstract Non-Interference and how it potentially provides new solutions to open problems in software security such as code injection and code obfuscation.*

Spring 2016 Refund Attacks on Bitcoin's Payment Protocol (Advanced Cryptography Course Project)

- *Presented a description and summary of real-life, experimentally verified attacks on the BIP70 Bitcoin Payment Protocol.*

Fall 2015 Efficient Proofs (Cryptography Course Project)

- *Presented a description and summary of the application of interactive proofs of knowledge and efficient zero knowledge proofs.*

Spring 2015 A Real-Time Adaptive Trading System Using Genetic Programming (Machine Learning in Quantitative Finance Course Project)

- *Presented a description and summary of applying genetic programming techniques to a real-time trading system.*

Inferring Features from Student Interactions with Educational Courseware (Big Data Course Project)

- *Analyzed data sets from the PSLC (Pittsburgh Science Learning Center) in order to determine whether other machine learning techniques could perform better at predicting student performance.*

Spring 2014 Predicting Power Consumption Using Linear Genetic Programming (Artificial Intelligence Course Project)

- *Used the Weka machine learning software to compare the effectiveness of genetic programming to other techniques (i.e. linear regression) at predicting the hourly electric power consumption of a building.*

Simple Web Server and Mail Client (Computer Networks Course Assignments)

- *Wrote a program to handle single HTTP requests and return HTTP responses to clients in Python.*
- *Wrote a program to communicate with a mail server using SMTP and send an email in Python.*

Fall 2013 Hardness Amplification and Error Correcting Codes (Theoretical Computer Science Course Project)

- *Presented a description and summary of methods to find functions that are hard to compute on the "average" instance, not just the worst case, including error correcting codes (i.e. the Walsh-Hadamard code, the Reed-Solomon code, Reed-Muller codes, and concatenated codes).*

Earlier Comparing Semi-Supervised and Supervised Machine Learning Algorithms on a Car Evaluation Data Set (Masters Project)

- *Compared the effectiveness of various semi-supervised machine learning algorithms (self-training, co-training, and transductive SVMs) to supervised algorithms (Naive Bayes and Tree Augmented Naive Bayes), each implemented in Java code, using a car evaluation data set.*