CPSC 490 Project Proposal:

Using Predicate Logic and LLMs to Assess Compliance with the Computer Fraud and Abuse Act

Rosie Rothschild

Faculty advisor: Ruzica Piskac

May 2, 2024

1 Introduction

While computer science is governed by the laws of code, the real world is governed by our legal code. This thesis builds upon methods of translation from legal code to predicate logic to determine whether situations are compliant with the US legal code. Specifically, it explores the Computer Fraud and Abuse Act (CFAA), which is a section of the US legal code that makes it illegal to "intentionally access a computer without authorization or exceeds authorized access" (Section 18 U.S. Code § 1030). [CFA] With hacking and cyber security breaches becoming more common, having an enforceable legal code to prosecute this abuse is essential.

2 Background

Congress enacted the Computer Fraud and Abuse Act in 1986 to federally prosecute crimes of hacking to obtain government information. The law has been amended in 1989, 1994, 1996, 2001, 2002, and 2008 to keep up with modern technology and broaden the scope of illegal conduct. Additionally, the Supreme Court recently specified the meaning of "exceeds authorized access" in *Van Buren v. United States* after a decades-long split among the circuits. With this new clarification of the law, the CFAA became more narrowly defined, applying only to instances where a person enters "particular areas of the computer– such as files, folders, or databases– that are off limits to him." (*Van Buren v. US*) [van21]. The court goes on to explain that liability under the CFAA "stems from a gates-up-or-down inquiry– one either can or cannot access a computer system, and one either can or cannot access certain areas within the system." (*Van Buren v. US*). So, one can only violate the CFAA if someone trespasses through a closed gate. (Kirr) [OK21] [PLB16]. Practically, it can be difficult to determine what falls under the CFAA, as the Supreme Court does not specify what a closed gate looks like. By reimagining the CFAA, a law utilized to regulate code, as a predicate logic code itself, we can better

understand what situations are compliant with this statute, and what situations would violate it.

There has been a body of work done to translate this legislation into predicate logic to apply to real-life situations (Verhij)[BV03]. The law is made up of argumentation schemes that are context-dependent and contingently valid, making it difficult to translate to an objective code. Verheij presents a method to formally model these schemes. Some work has been done building upon this model to formalize private international law and contracts (Minh Dung) [PMD09] [PMD10]. There has been no work on applying legal formalized logic to the CFAA, or more complex laws that contain cross-referencing. My thesis will explore mechanisms to translate this complex legal code, which will help to build the foundation of translation for even more complicated sections of the law in the future.

3 Problem Statement

This thesis will solve the following two problems: translating complex legal code into predicate logic and determining the application of the CFAA.

The CFAA is a complex law that governs cybercrime in the United States. It can be difficult to apply in various instances, and the penalties can change based on subtle distinctions in situations. This thesis formalizes the law in predicate logic to determine how to objectively apply the statue depending on the facts of a case.

There has yet to be a translation from CFAA to first-order logic, or any legal code with this much complexity. Since the CFAA includes cross-referencing within the section, it will require a non-trivial implementation of First Order Logic. This will be a step toward translating more complicated sections of law.

4 Methods and Planned Approach

This project will build upon existing methods to translate legal code to predicate logic. After further researching the background of CFAA, my approach will be to make a first-order logic dataset converting the legal code into predicate logic. This translation can be tedious and difficult, so researchers have begun to use Large Language Models (LLMs) to quickly perform this formalized translation. My approach will take advantage of GPT's powerful ability to recognize patterns to accomplish this task effectively. Then, I will clean and finalize the data to ensure it's in a usable format.

After successfully obtaining the translation from legal code to formalized logic, I will use SMT solvers to determine whether the given input is compliant with the CFAA. If the given input does not fit the compliance criteria, my project will determine what constraints are violated and how to make adjustments. Finally, I will create a GUI to interface with the backend SMT solver to create a user-friendly experience exploring the CFAA.

Additionally, because this is an Electrical Engineering and Computer Science joint thesis, there will be a brief literature analysis of the power utilized in LLM translation step.

5 Deliverables

1. A data set of input/output pairs of legal text and their corresponding formalized logic
2. App to check compliance with CFAA including:
   (a) backend logic using SMT solvers to determine if a situation violates the CFAA using translated predicate logic of CFAA
   (b) user-friendly GUI to test which situations violate the CFAA
3. Github Repository with code for app
4. Brief literature analysis of power utilization of hardware in LLM
5. Project Reports:
   (a) Project Proposal Presentation by 2/16
   (b) Midterm Progress Report by 3/29
   (c) Project Report Draft by 4/19
   (d) Project Poster by 4/26
   (e) Final Project Report by 5/02

6 Timeline

Date	Milestone
Weeks 1/15 – 1/22	Begin meeting with advisor, brainstorm topics
Week 1/29	Finalize topic, write project proposal
2/1	**Project Proposal Due**
Week of 2/5	Familiarize myself with LLM, SMT solvers, and the CFAA
Week of 2/12	Begin translation process from CFAA to predicate logic using LLM
2/16	**Project Proposal Presentation**
Week of 2/19	Formalize translation from data to predicate logic using LLM
Week of 2/26	Use formalized CFAA dataset to determine if a given situation is compliant with the CFAA. Implement logic to determine what statement is violated if not compliant
Week of 3/4	Finalize this implementation
Spring Break	**Catch Up time**
Week of 3/25	Work on Project Report and ensure backend logic works
3/29	**Midterm Progress Report**
Week of 4/1	Begin work on GUI
Week of 4/8	Debug and Test App; ensure front and backend work
Week of 4/15	Hardware Analysis of LLM Power Use, Work on final report
4/19	**Final Report First Draft Due**
Week of 4/22	Finalize Report and App
4/26	**Poster Presentation**
5/2	**Final Report Due**

References

[BV03] Bart Verheij. Dialectical Argumentation with Argumentation Schemes: An Approach to Legal Logic. *Artificial Intelligence and Law*, 2003.

[CFA] Section 18 U.S. Code § 1030.

[OK21] Orin Kirr. The Supreme Court Reigns in the CFAA in Van Buren. *Lawfare*, 2021.

[PLB16] Patricia L. Bellia. A Code-based Approach to Unauthorized Access Under the CFAA. *George Washington Law Review*, 1442, 2016.

[PMD09] Phan Ming Dung. Modular argumentation for modeling legal doctrines in common law of contract. *Artifical Intelligence and Law*, 2009.

[PMD10] Phan Ming Dung. A Logical Model of Private International Law. *Proceedings on the 10th International Conference on Deontic Logic in Computer Science*, pages 229–246, 2010.

[van21] Nathan Van Buren v. United States. 2021.