Project 4 Report

Adrian Hoang

April 11, 2025

1 Introduction

The primary objective was to explore the application of Large Language Models in the formalization of legal text, specifically translating the US tax code definition of a "dependent" (26 USC § 152(a)-(d)) into the Satisfiability Modulo Theories (SMT) formalism. The work involved several stages: leveraging an LLM for the initial translation to SMT-LIB 2 format, generating relevant test scenarios (compliant, non-compliant, and borderline cases) using an LLM, translating these scenarios into SMT assertions, verifying the logical consistency and compliance of these scenarios using the Z3 SMT solver, and iteratively refining the SMT model based on the results and collaboration.

2 Part 1: Legal Text to SMT Translation

The foundation of this project was the translation of the specified legal text, 26 USC § 152(a)-(d), into a formal SMT representation. This section describes the methodologies employed, the interaction with the LLM, and the refinement process leading to the final SMT model (submitted as `model.smt2`).

2.1 LLM Interaction Strategy & Techniques

To explore different approaches as per the assignment guidelines, we employed distinct strategies for interacting with the LLM (Gemini 2.5 through Google AI Studio):

- **(Chain-of-Thought - CoT):** Focused on prompting the LLM to first articulate the logical steps and breakdown of the law before attempting the SMT translation. The intention was to guide the LLM through a reasoning process mimicking human analysis.
- **(Structured Output):** Aimed at having the LLM first parse the legal text into an intermediate structured format (like a pseudo-JSON outline) before translating that structure into SMT. This approach sought to separate the legal interpretation from the formal language generation.

The following prompts exemplify these strategies:

2.1.1 CoT Prompt

```
1 You are an expert in logic and formal methods. Your task is to translate the
      following US tax code section, 26 USC 152(a)-(d), into the SMT-LIB 2 format
      for use with the Z3 solver.
2
```

```
Legal Text:
[... Inserted 26 USC 152(a)-(d) text here ...]

Instructions:
1. First, analyze the text subsection by subsection ((a), (b), (c), (d)). For each subsection, break down the conditions and rules into clear logical statements (like "An individual is a dependent IF they are a qualifying child OR a qualifying relative, AND no exceptions apply").
2. Identify the key entities (e.g., Taxpayer, Potential Dependent) and the properties or relationships involved (e.g., age, relationship type, residency, support provided, income, citizenship, filing status).
3. Define necessary SMT functions (`declare-fun`). Use Boolean (`Bool`) type for conditions and relationships where possible. Use Integer (`Int`) for age and potentially income/support thresholds if necessary, but prefer Booleans for simplicity if exact values aren't crucial for the logic structure. Assume two main constants `TP` (Taxpayer) and `PD` (Potential Dependent) of a custom sort `Person`.
4. Translate the logical breakdown from step 1 into SMT-LIB assertions (`assert`) using logical connectives (`and`, `or`, `not`, `=>`).
5. Pay close attention to the structure:
    *   Overall definition of dependent (a).
    *   Exceptions that disqualify someone (b).
    *   Definition of qualifying child (c) with its multiple conditions (A-E) and sub-rules (relationships, age, tie-breakers).
    *   Definition of qualifying relative (d) with its multiple conditions (A-D) and sub-rules (relationships, income, support, non-QC status).
6. Acknowledge any complex parts you are simplifying (e.g., tie-breaker rules in (c)(4), multiple support agreements in (d)(3), exact dollar amounts). Focus on capturing the main logical structure first.

Provide the SMT-LIB code.
```

Listing 1: Chain-of-Thought Prompt used for initial SMT generation

2.1.2 Structured Output Prompt

```
You are an expert in legal text analysis and formal representation. Analyze the provided US tax code section, 26 USC 152(a)-(d), defining a "dependent".

Legal Text:
[... Inserted 26 USC 152(a)-(d) text here ...]

Instructions:
1. First, extract the rules and conditions into a structured format (e.g., a hierarchical list or pseudo-JSON). Identify:
    *   The main concepts: Dependent, Qualifying Child, Qualifying Relative.
    *   The primary actors: Taxpayer (TP), Potential Dependent (PD).
    *   For each concept, list the necessary conditions (AND/OR relationships).
    *   List any exceptions that override the main rules.
    *   Identify key attributes needed for evaluation: relationship type, residency period, age, support fraction, gross income, citizenship/residency status, joint return filing.

2. Based *only* on the structured representation you created, translate these rules into SMT-LIB 2 format for Z3.
    *   Declare a sort `Person`. Declare constants `TP` and `PD` of sort `Person`.
```

```
16      *   Declare necessary functions (‘declare-fun‘) representing the attributes
        and conditions (mostly mapping ‘Person‘ or ‘Person Person‘ to ‘Bool‘ or ‘Int
        ‘). Examples: ‘(declare-fun IsQualifyingChild (Person Person) Bool)‘, ‘(
        declare-fun Age (Person) Int)‘, ‘(declare-fun ResidesWithTPMoreThanHalfYear (
        Person Person) Bool)‘.
17      *   Use ‘assert‘ statements to define the core logic (‘IsDependent‘, ‘
        IsQualifyingChild‘, ‘IsQualifyingRelative‘) based on the structured rules,
        using ‘and‘, ‘or‘, ‘not‘, ‘=>‘.
18      *   Clearly define the exceptions from section (b).
19      *   Simplify complex calculations (like exact support fractions or AGI
        comparisons in tie-breakers) using Boolean flags for now (e.g., ‘
        TPProvidesOverHalfSupport‘, ‘IsHighestAGIParent‘), noting the simplification.
20
21  Provide both the structured representation and the resulting SMT-LIB code.
```

Listing 2: Structured Output Prompt used for initial SMT generation

2.2 Analysis of LLM Outputs and Refinement

Comparing the initial outputs from both LLM prompting strategies proved valuable. The Chain-of-Thought approach often yielded more direct SMT code that attempted to capture the logical flow, but sometimes struggled with correctly nesting complex conditions or ensuring all predicates were properly defined and linked. The Structured Output approach generally produced a cleaner SMT skeleton based on the intermediate representation, promoting better organization. However, its accuracy was highly dependent on the quality of the initial structured extraction, and nuances could be lost in translation from the structure to SMT.

Both approaches initially presented common challenges typical of automated code generation from natural language. Frequently encountered issues included missing declarations for functions or constants, difficulty in precisely modeling nested definitions (like the various relationship types within the Qualifying Child and Qualifying Relative definitions), and accurately capturing complex logical constraints such as the requirement for a Qualifying Relative *not* being a Qualifying Child of any other taxpayer. Furthermore, translating inherently complex or underspecified rules, like the tie-breaker mechanisms in §152(c)(4) or the conditions for Multiple Support Agreements in §152(d)(3), required explicit simplification decisions.

The refinement process was iterative. We synthesized the strengths of both initial outputs, using the structured approach's organization while incorporating the logical connections highlighted by the CoT method. Significant manual correction was necessary, primarily adding missing declarations (`declare-fun`, `declare-sort`, `declare-const`) and ensuring correct SMT-LIB syntax. We carefully reconstructed the top-level logical structure to accurately reflect the law: a person is a dependent if and only if they are either a Qualifying Child or a Qualifying Relative, and none of the exceptions defined in §152(b) apply.

A crucial step involved defining a clear strategy for handling complexity. We decided to simplify certain aspects to focus on the core logic. Tie-breaker rules and multiple support agreements were abstracted using dedicated Boolean predicates . Thresholds like the gross income limit for Qualifying Relatives were initially modeled simply but later refined using integer types (`Int`) for accurate comparison, while support and residency fractions were represented by Boolean predicates (e.g., `TPProvidesOverHalfSupport`). The challenging "any other taxpayer" condition was modeled via a predicate (`IsQC_OfAnyOtherTaxpayer`) whose truth value would need to be asserted based on specific scenario context. Throughout this process, the Z3 solver was used repeatedly to check for syntax errors and logical inconsistencies, guiding the refinement towards the final, compilable `model.smt2` file.

2.3 Correctness Checks

Before proceeding, the validity of the `model.smt2` file was assessed. Firstly, it was confirmed that the file could be parsed and accepted by the Z3 SMT solver without syntax errors. Secondly, manual evaluation was performed using simple test cases. For instance, tracing the logic for a typical Qualifying Child (e.g., a 10-year-old child living with their parent, meeting support and filing requirements) confirmed the model correctly evaluated `IsQualifyingChild` to true and `IsDependent` to true. Conversely, a scenario involving an unrelated adult friend with significant income clearly failed both Qualifying Child and Qualifying Relative tests within the model, resulting in `IsDependent` evaluating to false. These checks provided confidence that the SMT model captured the essential logical structure of the statute, acknowledging the documented simplifications.

3 Part 2: Scenario Generation and Translation

To test the formalized SMT model, we required realistic scenarios reflecting different dependency outcomes.

3.1 LLM-Based Scenario Generation

We tasked the LLM with creating three distinct scenarios designed to test different paths through the dependency rules. The prompt guided the LLM to generate cases where the potential dependent clearly qualified, clearly did not qualify, and one where the status was borderline or ambiguous.

```
1 Based on the rules for determining a dependent in 26 USC 152 (Qualifying Child or
      Qualifying Relative, considering exceptions), please generate three detailed
      scenarios involving a Taxpayer (TP) and a Potential Dependent (PD). Provide
     enough detail about their relationship, age, residency, support, income,
     citizenship, and filing status to determine dependency.
2
3 1.  **Scenario 1: Clear Dependent:** Create a scenario where PD is clearly a
     dependent of TP.
4 2.  **Scenario 2: Clear Non-Dependent:** Create a scenario where PD is clearly *
     not* a dependent of TP.
5 3.  **Scenario 3: Borderline/Ambiguous:** Create a scenario where the dependency
     status is borderline or depends on a subtle aspect of the rules (e.g.,
     residency slightly over/under half the year, income close to the limit,
     meeting QR relationship only via household member rule, or potentially being
     a QC of another person).
```

Listing 3: LLM Prompt for Generating Test Scenarios

The LLM generated the following scenarios:

Scenario 1: Clear Dependent (Qualifying Child): Alice (TP, 40) and her son Bob (PD, 15). Bob lived with Alice all year, received full support from her, had no income, and didn't file a joint return. All citizenship and other basic requirements were met. This scenario was designed to clearly satisfy the Qualifying Child tests.

Scenario 2: Clear Non-Dependent: Charles (TP, 50) and his sister Diane (PD, 48). Diane lived separately, was fully self-supporting with a substantial income ($60,000), and received no support from Charles. This scenario was designed to fail multiple QC and QR tests (residency, support, income).

Scenario 3: Borderline (Qualifying Relative - Income Test): Eve (TP, 60) and her father Frank (PD, 80). Frank lived separately, but Eve provided over half his support. Frank's gross income ($4,800) was deliberately set slightly above the hypothetical exemption amount threshold ($4,700 for the year). This scenario tested the Qualifying Relative rules, specifically hinging on the gross income limitation.

3.2 Scenario Translation to SMT Formalism

Each generated scenario was then translated into a corresponding SMT-LIB 2 file. These files included the definitions from the base `model.smt2` (or assumed it was implicitly loaded) and added specific `assert` statements to represent the facts of the scenario. For example, in clear-dependent.smt2, assertions included '(assert (= TP Alice))', '(assert (= PD Bob))', '(assert (= (Age Bob) 15))', '(assert (IsChildOrDescendant Bob Alice))', '(assert (ResidesWithTPMoreThanHalfYear Alice Bob))', '(assert (not (PDProvidesOverHalfOwnSupport Bob)))', etc., carefully reflecting all details provided in the scenario description. These scenario files set up the specific context for verification by the SMT solver.

4 Part 3: Compliance Verification with SMT Solvers

With the formalized model (`model.smt2`) and scenario-specific assertions prepared, the Z3 SMT solver was employed to verify compliance. Each scenario file was executed using the command `z3 -smt2 <scenario_file.smt2>`.

The solver's output provided confirmation of the model's behavior under each set of facts:

4.0.1 Scenario 1 Verification Output (Clear Dependent)

Executing `z3 -smt2 clear-dependent.smt2` yielded:

```
sat
Scenario 1: Is Bob a dependent of Alice?
true
Is Bob a Qualifying Child of Alice?
true
Is Bob a Qualifying Relative of Alice?
false
Do Exceptions Apply?
false
```

Listing 4: Z3 Output for Scenario 1

Z3 reported `sat`, indicating consistency. Evaluating `(IsDependent Alice Bob)` yielded `true`, correctly identifying Bob as a dependent via the Qualifying Child rules.

4.0.2 Scenario 2 Verification Output (Clear Non-Dependent)

Executing `z3 -smt2 clear-nondependent.smt2` yielded:

```
sat
Scenario 2: Is Diane a dependent of Charles?
false
Is Diane a Qualifying Child of Charles?
false
Is Diane a Qualifying Relative of Charles?
```

```
false
Do Exceptions Apply?
false
```

Listing 5: Z3 Output for Scenario 2

Z3 reported `sat`. Evaluating `(IsDependent Charles Diane)` yielded `false`. Further evaluations confirmed that both `IsQualifyingChild` and `IsQualifyingRelative` were false, aligning with Diane failing multiple dependency tests.

4.0.3 Scenario 3 Verification Output (Borderline)

Executing `z3 -smt2 borderline.smt2` yielded:

```
sat
Scenario 3: Is Frank a dependent of Eve?
false
Is Frank a Qualifying Child of Eve?
false
Is Frank a Qualifying Relative of Eve?
false
Does Frank meet the QR Gross Income Test?
false
Do Exceptions Apply?
false
```

Listing 6: Z3 Output for Scenario 3

Z3 reported `sat`. Evaluating `(IsDependent Eve Frank)` yielded `false`. The crucial evaluation `(< (GrossIncome Frank) ExemptionAmountThreshold)` also yielded `false`, pinpointing the failure at the gross income test for Qualifying Relatives, thus correctly classifying Frank as non-dependent.

4.1 Discussion of Solver Output

In summary, the Z3 solver successfully verified the logical consistency of each scenario (`sat`) and correctly classified the dependency status according to the formalized rules in `model.smt2`. The results, detailed above, demonstrated that the SMT model accurately captured the core logic of the statute, including handling specific threshold conditions like the gross income test, for the tested cases. The solver's output provided precise confirmation of the model's behavior under different factual circumstances and validated the logic implemented in the SMT files.

5 Part 4: Model Refinement and Development Process

The journey from the initial LLM output to the final, verified SMT model involved several cycles of refinement, highlighting both the potential and the pitfalls of using LLMs for legal formalization.

The initial SMT code generated by the LLM, regardless of the prompting strategy, required manual intervention. Common issues included syntactic errors and semantic errors. Ensuring the correct logical structure, particularly the interplay between the main definition in §152(a), the exceptions in §152(b), and the definitions of Qualifying Child (§152(c)) and Qualifying Relative (§152(d)), required careful manual construction of the corresponding SMT definitions (`define-fun`). Translating nuanced phrasing, such as conditions involving negation ("has *not* provided over one-half") or specific thresholds, also demanded attention to detail.

An important part of the refinement involved addressing complexity. We made conscious decisions about which rules to simplify, such as the tie-breaker rules and multiple support agreements, representing them with abstract Boolean predicates. This allowed us to focus on verifying the core dependency logic. A significant refinement occurred when testing the borderline scenario (Scenario 3); initially, modeling income with Booleans proved insufficient. Switching the relevant predicates (`GrossIncome`, `ExemptionAmountThreshold`) to use the `Int` sort allowed for the precise numerical comparison required by the statute, leading to the correct evaluation by Z3. Another refinement involved correcting the order of declarations in scenario files, ensuring constants representing individuals were declared before being used in assertions.

Throughout this process, unexpected results from Z3 often signaled errors in the model. For instance, if a clear dependent case was evaluated as non-dependent, it usually pointed to a flaw in the logical structure of the `IsDependent`, `IsQualifyingChild`, or `IsQualifyingRelative` definitions, or incorrect application of the exceptions. Debugging involved tracing the logic manually and adjusting the SMT definitions until Z3 produced the expected output for known test cases.

The iterative cycle of translation, testing, analysis, and refinement was crucial in developing a model that, within its documented limitations, accurately reflects the target legal text. The final SMT model (`model.smt2`) and the scenario files represent the culmination of this process.