Tax situations as named rules:

a symbolic model for finding the lowest lawful tax

KnotTheory.ai, Inc.
Fremont, California

October 2026

Abstract

Most taxpayers could file their return in more than one lawful way, since the same facts allow different filing statuses, the standard or the itemized deduction, several depreciation methods and, for a taxpayer who is willing to act during the year, changes such as forming an S corporation or opening a retirement plan, yet preparation software fills in only the one return that the user describes. In this paper, we describe a model in which each provision of the tax law is written as a named rule in a small language, Bell-Tax, which grew out of Bell, the short-voice-prompt language of our Tesseract-Prime assistant. Because a rule may use other rules by name, a question such as "can this taxpayer file as head of household?" is answered by following the names down until every branch rests on a fact, and a fact that is not yet known becomes the next question in the interview. A named AI agent may be asked a question whose answer is itself the name of another rule, so that the agent picks the provision that governs and the rule then computes the amount. Elections and optional actions split the evaluation into scenarios, each of which fills the fixed set of IRS forms, is computed by a deterministic tax engine and is ranked by its total tax plus the cost of the action. For a sole proprietor in California with $150,000 of profit, the four scenarios computed by our production engine for tax year 2025 range from $46,591 down to $30,643 in federal and state tax, payroll tax and costs.

1 Introduction

Consider a software consultant in San Jose who works for herself. In 2025 she billed $180,000 and spent $30,000 running her business, so her Schedule C shows $150,000 of profit, and her preparer, or the software she uses, asks for these numbers and produces a correct return. It is worth noting, however, that this return is only one of several correct returns she could have filed. Had she formed a limited liability company early in the year and elected to be taxed as an S corporation, part of her profit would have been paid to her as salary and the rest would have flowed to her as a distribution, and her payroll tax would have been quite different. Had she opened a SEP-IRA before the filing deadline, she could have put away about a fifth of her net earnings and deferred the tax on it. She could also have done both, or given next year's charitable gifts this year. As we will see in Section 6.4, the gap between the return she filed and the best of these alternatives is about $16,000 in a single year.

None of this is news to a good preparer, because the Internal Revenue Code gives the taxpayer choices on purpose. A married couple may file jointly or separately, a filer may take the standard deduction or itemize, and a business owner may expense an asset under §179, take bonus depreciation or depreciate it over its recovery period. Each of these choices is lawful, each one changes the tax, and they interact with one another in ways that are tedious to work out by hand. Note

that the set of forms that can be filed with the IRS is fixed, so what changes from one choice to another is which of those forms are used and what is written on them.

Preparation software, whether it is used by a taxpayer or by a certified public accountant (CPA), asks for the facts and fills in a return. Some products compare two filing statuses or show a "what if" for a change that the preparer types in, and planning tools let a professional model a few alternatives by hand. However, none of them starts from the facts and lists the lawful alternatives on its own. Hence the lowest tax that a taxpayer could lawfully pay is found, when it is found at all, through the experience and diligence of the preparer.

In this paper, we describe a model that carries out this search. Let a *tax situation* be the facts of one taxpayer for one year: who the taxpayer is, who depends on them, what they earned and from whom, what they own and owe, what they gave away, and what they answered in the interview. Let a *rule* be a named statement of one provision of the law, giving when it applies, what it changes, its limits and its citation. The model writes every rule for a year in a small language, matches the taxpayer's situation against all of the rules, and produces every lawful scenario as a complete return. The scenarios then pass through a set of anti-abuse checks, and the one with the lowest total cost is shown to the user along with every rule that produced it.

Two properties of the language carry most of the design, and we will come back to both of them throughout the paper. The first is that a rule has a name, and a name can be used wherever a value can be used. Thus a rule may refer to another rule by name, in a condition or in a computation, and evaluation follows those names until every branch rests on a fact. This is how logic programming languages such as Prolog evaluate a query, and it has a pleasant side effect for tax: a fact that is needed and not known is exactly the question to ask the user next. The second property is that a named AI agent may be asked a question whose answer is the name of another rule. For example, a model reads a receipt and decides that it is a business meal, and the rule for business meals then computes the 50% that is deductible. Intuitively, the model is trusted with the judgment it is good at, and the arithmetic is left to a rule that can be read and checked.

We should be clear about the scope of this work. The strategies that the model finds, such as an S corporation election, a SEP-IRA or the bunching of charitable gifts, are well known to practitioners, and we make no claim on them. Our contribution is the representation and the machinery that find such strategies from the facts, check them against the law and produce the returns.

The rest of the paper is organized as follows. Section 2 reviews prior work, Section 3 describes Bell, the language we started from, and Section 4 describes the extensions that make up Bell-Tax. Section 5 explains how the model evaluates the rules and enumerates the scenarios, and Section 6 walks through five worked examples, each of which was run end to end. Section 7 looks at other domains in which the same machinery applies, and Section 8 discusses its limits.

2 Prior work

Writing law as a program has a long history. Sergot et al. [1] wrote the British Nationality Act 1981 as a Prolog program in 1986, and Sherman [2] modelled the Income Tax Act of Canada in Prolog the following year. Lawsky [3] gave a logic for statutes that treats defaults and exceptions explicitly. Catala [4] is a programming language designed for the literate translation of tax law, and it has been used for the French family benefits code, while OpenFisca and PolicyEngine compute taxes and benefits from declarative parameters for microsimulation. The IRS built Direct File on the *Fact Graph* [5], a declarative graph of facts and derived values written in XML and Scala that reasons about incomplete information, and published it in 2025. Intuit's patents describe tax rules

as completeness graphs and calculation graphs [6].

Each of these systems computes the one return that follows from a given set of facts and choices. Planning software lets a professional compare scenarios that the professional sets up by hand, and United States Patent No. 7,970,683 [7] restructures a group of corporate entities to minimize multi-state tax by combining, shuffling and splitting entities under an optimization algorithm. To our knowledge, none of these writes the elections and the optional actions themselves as rules, enumerates the scenarios from the taxpayer's facts, lets an AI agent choose among rules by name, or produces a complete set of forms for each scenario.

Several of the individual parts are old, and we would like to acknowledge them. Sergot's query-the-user facility [8] let a logic program ask the user for a fact that it could not derive, and expert systems have done the same ever since. Language-model routers, such as the router chains of the LangChain library, let a model choose which prompt runs next, and function calling restricts a model's answer to an enumerated set of values. Answer set programming enumerates the models of a logic program and can minimize an objective over them, and the goal-directed s(CASP) system has been applied to legal reasoning [9]. Logic-LM [10] and later work translate problems, including sections of the Internal Revenue Code, into programs for a symbolic solver. What we have not found is the combination of four things: a model's answer that names a rule and continues the same resolution while being recorded as an assumption; options with costs that change the facts of a scenario; every scenario computed into a complete return by a deterministic engine; and a test of which assumptions the ranking actually depends on.

Section 14 of the Leahy–Smith America Invents Act deems any strategy for reducing, avoiding or deferring tax liability to be within the prior art, with an exception for a method or system used solely for preparing a tax return. We read that section as a statement of where a contribution can lie in this area, namely in the language and the machinery, and that is where we have put our effort.

3 Bell

Bell is the command language of Tesseract-Prime, our desktop assistant. A user records a *short voice prompt*, which is a short name such as "morning" or "show xcode" bound to a text, and when the name is spoken the assistant looks it up and acts on its text. The text may be a command, a chain of further names that the assistant walks in order, or plain language, in which case it is sent to a language model as a prompt; a text beginning with `llm_function_` calls a native function instead. A Bell program composes these steps:

```
PAUSE,
VERB "hello",
MIC,
VAR gift = VALUEOF("world"),
IF (LOOKFOR("my text") == yes) THEN { VERB "world", VERB "goose" } ELSE { VERB "hello" }
END
```

The grammar is written for the Algodal parser generator, which emits a C parser that produces an abstract syntax tree, and the assistant then executes the tree. Two ideas from Bell turned out to be exactly what tax needs. The first is that a program has a name, so that a name can be used wherever a step can and programs are built from earlier programs simply by naming them; a user who has recorded "open mail" and "read inbox" can record "morning" as those two names in a row. The second is that a step which is not a command is treated as a question for a model. Bell-Tax keeps both of these ideas and adds what a tax language requires, namely typed facts,

arithmetic with limits and phase-outs, parameters, elections, options with costs, loops, guards and an objective.

4 Bell-Tax

In this section we describe the language one construct at a time. The listings are short extracts from the rule pack that we use in Section 6, so the reader will meet them again there in context.

4.1 Facts and rules

A *fact* is a value at a path in the tax situation. For example, `FACT return.schedule_c_net` is the profit on Schedule C, and `FACT p.months_lived_with_taxpayer` is the number of months that a dependent `p` lived at home. Facts come from three places, namely the documents the taxpayer uploads (a W-2, a 1099, a receipt), the answers to interview questions, and the results of earlier rules. A *rule* is a named program with optional parameters and one or more citations, and its body is a value, a condition or a block of statements:

```
RULE child_age_test (p) CITE "IRC 152(c)(3)"
  = FACT p.age < 19 OR (FACT p.age < 24 AND FACT p.full_time_student == YES)
    OR FACT p.permanently_disabled == YES
END

RULE qualifying_child (p) CITE "IRC 152(c)"
  = child_relationship(p) AND child_age_test(p) AND child_residence_test(p)
    AND NOT FACT p.provided_over_half_own_support == YES
    AND NOT FACT p.filed_joint_return == YES
END

RULE salt_cap CITE "IRC 164(b)(6) as amended by P.L. 119-21"
  = MAX(PHASEOUT(40000, FACT return.magi, 500000, 30%), 10000)
END
```

The first rule reads almost as the statute does, since a child meets the age test if the child is under 19, or under 24 and a full-time student, or permanently disabled. The second rule refers to three other rules by name. A bare name such as `child_age_test(p)` inside a condition is a rule used as a predicate, and a reader can look it up in the same way that a reader of the Code looks up a cross-reference. The third rule shows how a 2025 amendment is written: the cap on the deduction for state and local taxes is $40,000, reduced by 30% of modified AGI above $500,000, but it never falls below $10,000.

The rules of one tax year form a *rule pack*, which begins with `YEAR 2025 JURISDICTION "US"`. Since most of the Code does not change from one year to the next, a pack may declare `INHERITS 2024` and restate only the provisions that changed. State packs are written in the same way, and a California pack, for example, declines the federal §199A deduction simply by not containing it.

4.2 Recursion to facts, and the interview

Evaluating a named rule is called *resolution*. To decide `head_of_household_eligible`, the resolver evaluates its conditions, one of which names `any_qualifying_person`, which counts `qualifying_child(p)` over the dependents, which in turn names three tests, each of which reads facts. The recursion ends when every branch has reached a fact. If a fact is missing, resolution stops and returns the `ASK` attached to that fact, i.e. the question to put to the user:

```
RULE ask_months_lived (p) CITE "IRC 152(c)(1)(B)" {
  ASK p.months_lived_with_taxpayer "How many months of 2025 did this person live with you?"
} END
```

Thus the interview is generated by the rules themselves. It consists of the facts that resolution actually reaches for this particular taxpayer, asked in the order in which they are reached, and a question that no rule needs is never asked. For example, a parent of two adult children who are not students is never asked where the children lived, because the age test fails before the residence test is reached. Resolution uses memoization, so that a rule which has been evaluated once with given arguments is not evaluated again. A rule that reaches itself through its own names with the same arguments forms a cycle, and the resolver reports the cycle as an error in the rule pack.

4.3 Agents, and answers that are rules

Some facts are judgments, such as whether a receipt is a business expense, what a reasonable salary is for an occupation, or whether a change has a purpose apart from tax. A language model is good at judgments of this kind, and it is unreliable at arithmetic that nobody checks. Bell-Tax therefore declares named agents and calls them, or calls a model directly, as values:

```
AGENT receipt_reader
  MODEL "vision-language model, zero data retention"
  ROLE "Read one receipt or invoice and say which provision governs it."
  TOOLS (document_text, document_image)
END

RULE receipt_treatment (r) CITE "IRC 162; IRC 262; IRC 274(n); IRC 263(a); IRC 280A"
  = AGENT(receipt_reader, "Which of these provisions governs this receipt?", FACT r.text)
    EXPECT RULE IN (business_meal, business_supply, home_office_cost, rental_repair,
                    rental_improvement, personal_expense) CONFIRM
END

RULE business_meal (r) CITE "IRC 274(n)(1)" = 50% * FACT r.amount END
```

With `EXPECT RULE`, the agent's answer is the name of another rule, and the resolver continues by resolving that rule with the same inputs. The answer must be one of the rules in the list, and an answer outside the list is discarded and the question is put to the user instead. Every agent answer is recorded as an *assumed* fact together with its provenance, that is, which agent and which model answered, what question was asked and what inputs were given. `CONFIRM` marks the answer for the user to accept before the return is final. Note that a model never writes an amount on the return; its role is limited to choosing a rule, or supplying an assumed fact that a rule then uses.

Since the rule an agent chooses may itself call an agent, a chain of choices and resolutions can run to any depth, and it ends, as every resolution does, at facts. In Section 6.5 we show an identification key from botany that works exactly this way, with one agent question at each step.

4.4 Elections, options and costs

The law gives the taxpayer two kinds of choice, and Bell-Tax has a construct for each kind. An `ELECT` is a choice made on the return itself, such as the standard or the itemized deduction, and each `CHOICE` under it is a branch. An `OPTION` is an action that the taxpayer could take in the world, such as forming a company. An option has a cost, and the tax it saves may be permanent or may only be postponed to a later year, so the language records both the cost and any deferral:

```
RULE scorp_option CITE "IRC 1361; IRC 1362; IRC 1366; IRC 3111" {
  IF FACT return.schedule_c_net > 60000 AND FACT taxpayer.entity_type == "none" THEN {
    LET market = AGENT(wage_researcher, "Median annual wage for this occupation and metro area",
                   FACT taxpayer.occupation, FACT taxpayer.metro_area) EXPECT money CONFIRM
    LET salary = MAX(market, 40% * FACT return.schedule_c_net)
    LET employer_fica = 7.65% * salary
    LET k1 = FACT return.schedule_c_net - salary - employer_fica - 42 - 1200
    OPTION form_llc_elect_s "Form an LLC and elect S-corporation status (Form 2553)"
      COST employer_fica + 42 + 1200 {
      SET FORM "W-2" LINE "1" = salary
      SET FORM "Schedule K-1 (1120-S)" LINE "1" = k1
      SET FACT return.schedule_c_net = 0
      SET FACT return.k1_final = k1
      SET FACT taxpayer.entity_type = "s_corp"
      USE form_1120s
    }
  } END
} END
```

It is worth reading this rule line by line. The option is offered only to a sole proprietor with more than $60,000 of profit, since below that level the payroll savings rarely cover the cost of running a corporation. The wage agent is then asked for the market wage of the occupation in the taxpayer's metro area, and its answer is marked `CONFIRM`. The salary is taken as that wage or 40% of profit, whichever is larger. The employer's share of payroll tax, a $42 federal unemployment tax and $1,200 of annual administration together make up the cost of the option, and whatever profit remains flows to the owner on Schedule K-1.

`SET FORM` writes a line of the scenario's return, while `SET FACT` changes the scenario's facts, and the second matters more than it might first appear. Once the business is an S corporation it has no Schedule C, so any rule that reads `schedule_c_net`, such as the SEP-IRA option, sees zero in that scenario and is not offered there. California's tax on the S corporation is written as a separate rule, `ca_scorp_tax`, because it depends on the K-1 after every option that changes the K-1 has been applied; we list it with the others in Section 6.4.

4.5 Loops

Bell-Tax has three loops. `FORALL p IN taxpayer.dependents` iterates over a collection, `FOR y FROM 2025 TO 2027` counts, for plans that span several years, and `REPEAT {...} UNTIL` *condition* `LIMIT` *n* runs a block until a condition holds. The limit is required, so that every program terminates. One might ask why a tax language needs a loop that runs until convergence, and the answer is that the law itself contains a circular computation. The self-employed health insurance deduction lowers modified AGI, which raises the premium tax credit, which in turn lowers the premiums that are left to deduct. Rev. Proc. 2014-41 resolves this circularity by iteration, and the rule does the same:

```
RULE sehi_and_ptc CITE "IRC 162(l); IRC 36B; Rev. Proc. 2014-41 sec. 4" {
  LET d = 0
  REPEAT {
    LET ptc = MIN(premium_tax_credit(FACT return.agi_before_sehi - d), FACT return.premiums)
    LET d_next = MIN(FACT return.premiums - ptc, FACT return.schedule_c_net - FACT return.se_tax /
         2)
    LET delta = d_next - d
    LET d = d_next
  } UNTIL delta < 1 AND delta > -1 LIMIT 25 END
```

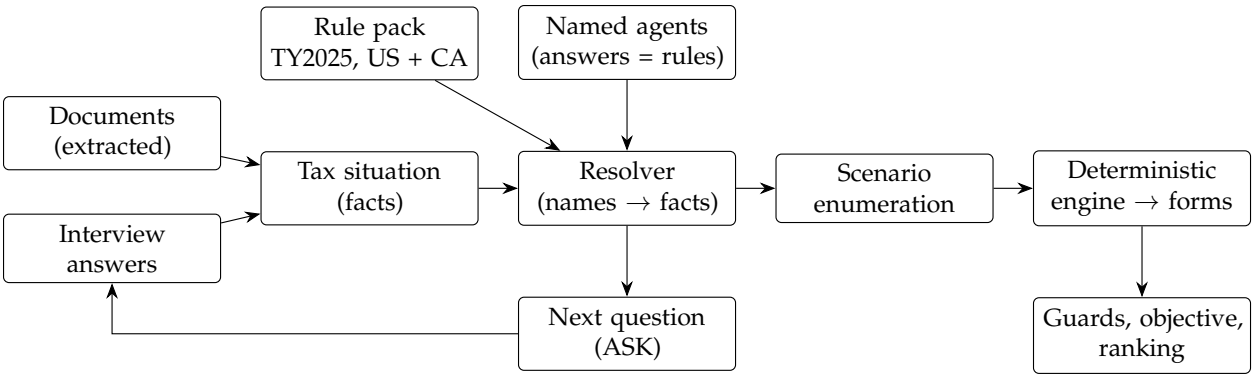


Figure 1: The model. Facts and a rule pack go to the resolver, which asks for missing facts; the scenarios it enumerates are computed by the engine into form sets and ranked.

```
  SET FORM "Schedule 1" LINE "17" = d
  SET FORM "8962" LINE "24" = ptc
} END
```

4.6 Guards and the objective

A `GUARD` removes a scenario that would not survive examination, and it carries a citation and a reason that is shown to the user. For example, one guard rejects an S corporation salary that is below a defensible share of profit (§3121(a), Rev. Rul. 74-44), and others reject a change with no purpose apart from tax (§7701(o)) and an activity not engaged in for profit (§183). The `OBJECTIVE` states what "best" means for this taxpayer:

```
OBJECTIVE MINIMIZE FACT scenario.federal_tax + FACT scenario.state_tax + FACT scenario.cost
  OVER 3 YEARS DISCOUNT 5%
END
```

Money that is put into a retirement plan is taxed when it is taken out, so counting a deferral as a saving would flatter the retirement options. The model therefore carries deferrals separately and values them over the horizon at an assumed future tax rate, and the user sees the one-year total and the deferred amount side by side.

5 Evaluation

Figure 1 shows the flow from documents to a ranked list of returns. The resolver evaluates a set of *entry rules* over the facts, namely the filing status, the deduction and each of the options. When it reaches an `ELECT` or an `OPTION` whose branch has not yet been decided, it forks, and each branch becomes a scenario that is evaluated on its own copy of the facts, so that a change made in one scenario is never seen by another. Three questions then arise, which we take in turn: in what order the rules should run, how to keep the number of scenarios small, and how a scenario becomes a return.

Order. Let $R(r)$ and $W(r)$ be the sets of fact paths that a rule r reads and writes, taken through every rule that r names. If one rule writes a path that another rule reads, the writer must run first, and the resolver orders the entry rules accordingly, in the way that stratified Datalog orders its rules. Hence the condition of an option is tested on the facts as they stand when the option is

reached, after the effects of the options before it have been applied. In the example of Section 6.4, the S corporation option writes `schedule_c_net` and the SEP-IRA option reads it, so the S corporation option runs first, and in every scenario that forms the S corporation the SEP-IRA is not offered. If the write–read order contains a cycle, the resolver reports it as an error in the rule pack.

Size. With k binary choices there are up to 2^k scenarios, a number that grows quickly. However, two choices interact only if one of them writes something that the other reads, either directly or through the engine's computation, and a choice that interacts with nothing can be settled inside each scenario by simply taking the better branch. The standard or itemized deduction is the usual example of such a choice, because once every option that changes an itemized amount has been decided, the deduction can be chosen locally. If j of the k choices are independent in this sense, the count falls from 2^k to 2^{k-j}, and in the example of Section 6.4 sixteen branches become eight.

Returns. Each scenario's `SET FORM` lines and facts are handed to the deterministic engine that computes returns in production at Tesseract Tax Cloud. The engine fills the fixed set of federal and state forms and returns the tax, so every scenario is a complete return that could be filed as it stands. The guards are then applied, the objective is evaluated and the scenarios are ranked, and the user sees the best scenario, the ones close to it, and for each of them the trace of rules, citations and assumed facts that produced it.

6 Worked examples

The examples in this section were run, and the outputs shown are copied from that run. A script reads the example rule packs with the Bell-Tax reference parser, the resolver evaluates them over the facts shown, and for the scenarios of Section 6.4 every `SET FORM` line is handed to the Tesseract Tax Cloud engine, which computes the tax under 2025 law. The rule listings are copied verbatim from the rule packs. So that the run can be repeated, the agent answers are supplied by fixed stand-ins, whereas in production they come from the named agents. In the traces, indentation shows the depth of resolution, the bracket after a rule is its citation, and a line marked *assumed* is an agent answer recorded with its provenance.

6.1 Resolution, and the interview it generates

Consider a single parent who paid more than half the cost of keeping up the home and who has two children. The elder, Asha, is 22 and is not a student. For the younger, Ravi, who is 16, the number of months he lived at home is not yet known. We would like to know whether the parent can file as head of household, and the rules involved are:

```
RULE head_of_household_eligible CITE "IRC 2(b)"
  = FACT taxpayer.married_at_year_end == NO
    AND FACT taxpayer.paid_over_half_home_cost == YES
    AND any_qualifying_person
END

RULE any_qualifying_person CITE "IRC 2(b)(1)(A)"
  = MAX(VALUEOF("count_qualifying_children"), 0) > 0
END

RULE count_qualifying_children CITE "IRC 152(c)" {
  LET n = 0
  FORALL p IN taxpayer.dependents {
    IF qualifying_child(p) THEN { LET n = n + 1 } END
  }
```

```
} END

RULE qualifying_child (p) CITE "IRC 152(c)"
  = child_relationship(p) AND child_age_test(p) AND child_residence_test(p)
    AND NOT FACT p.provided_over_half_own_support == YES
    AND NOT FACT p.filed_joint_return == YES
END

RULE child_relationship (p) CITE "IRC 152(c)(2)"
  = FACT p.relationship == "child" OR FACT p.relationship == "stepchild"
    OR FACT p.relationship == "foster_child" OR FACT p.relationship == "sibling"
    OR FACT p.relationship == "descendant"
END

RULE child_age_test (p) CITE "IRC 152(c)(3)"
  = FACT p.age < 19 OR (FACT p.age < 24 AND FACT p.full_time_student == YES)
    OR FACT p.permanently_disabled == YES
END

RULE child_residence_test (p) CITE "IRC 152(c)(1)(B)"
  = FACT p.months_lived_with_taxpayer > 6
END

RULE ask_months_lived (p) CITE "IRC 152(c)(1)(B)" {
  ASK p.months_lived_with_taxpayer "How many months of 2025 did this person live with you?"
} END
```

The query is `head_of_household_eligible`. Resolution follows the names down until it needs a fact that it does not have, and then it stops and returns the `ASK` attached to that fact. The answer is stored, and the query is resolved again:

```
?- head_of_household_eligible

head_of_household_eligible()  [IRC 2(b)]
  any_qualifying_person()  [IRC 2(b)(1)(A)]
    count_qualifying_children()  [IRC 152(c)]
      qualifying_child(Asha)  [IRC 152(c)]
        child_relationship(Asha)  [IRC 152(c)(2)]
        child_age_test(Asha)  [IRC 152(c)(3)]
      qualifying_child(Ravi)  [IRC 152(c)]
        child_relationship(Ravi)  [IRC 152(c)(2)]
        child_age_test(Ravi)  [IRC 152(c)(3)]
        child_residence_test(Ravi)  [IRC 152(c)(1)(B)]

unknown fact  Ravi.months_lived_with_taxpayer
ASK           How many months of 2025 did this person live with you?

answer: 12

head_of_household_eligible()  [IRC 2(b)]
  any_qualifying_person()  [IRC 2(b)(1)(A)]
    count_qualifying_children()  [IRC 152(c)]
      qualifying_child(Asha)  [IRC 152(c)]
        child_relationship(Asha)  [IRC 152(c)(2)]
        child_age_test(Asha)  [IRC 152(c)(3)]
      qualifying_child(Ravi)  [IRC 152(c)]
        child_relationship(Ravi)  [IRC 152(c)(2)]
        child_age_test(Ravi)  [IRC 152(c)(3)]
        child_residence_test(Ravi)  [IRC 152(c)(1)(B)]

head_of_household_eligible = YES
```

It is instructive to notice what was not asked. Asha fails the age test at `child_age_test`, so no further fact about her is read, and in particular nobody asks where she lived. For Ravi, the residence test needs the one fact that is unknown. Exactly one question is asked, and it is asked because a rule needed its answer; on the second pass Ravi qualifies, and hence so does the parent.

6.2 An agent answer that is a rule

Next, a receipt-reading agent is asked which provision governs each receipt. Its answer must be one of six rule names, and the rule that it names computes the amount:

```
AGENT receipt_reader
  MODEL "vision-language model, zero data retention"
  ROLE "Read one receipt or invoice and say which provision governs it."
  TOOLS (document_text, document_image)
END

RULE receipt_treatment (r) CITE "IRC 162; IRC 262; IRC 274(n); IRC 263(a); IRC 280A"
  = AGENT(receipt_reader, "Which of these provisions governs this receipt?", FACT r.text)
    EXPECT RULE IN (business_meal, business_supply, home_office_cost, rental_repair,
                    rental_improvement, personal_expense) CONFIRM
END

RULE business_meal (r) CITE "IRC 274(n)(1)" = 50% * FACT r.amount END

RULE rental_improvement (r) CITE "IRC 263(a); IRC 168(c); Rev. Proc. 87-57 Table A-6"
  = FACT r.amount * TABLE("macrs_residential_27_5_first_year", FACT r.month_placed_in_service)
END

RULE personal_expense (r) CITE "IRC 262" = 0 END
```

We give the resolver three receipts, namely a client dinner, a new roof on a rental house placed in service in July, and a grocery bill. The result is:

```
?- receipt_treatment("Dinner with client, Acme Corp", 180.00)
receipt_treatment(Dinner with client)  [IRC 162; IRC 262; IRC 274(n); IRC 263(a); IRC 280A
    ]
  assumed (AGENT receipt_reader, to confirm): Which of these provisions governs this recei
      ... -> business_meal
  business_meal(Dinner with client)  [IRC 274(n)(1)]
= 90.00

?- receipt_treatment("New roof, 12 Elm St rental", 14,000.00)
receipt_treatment(New roof)  [IRC 162; IRC 262; IRC 274(n); IRC 263(a); IRC 280A]
  assumed (AGENT receipt_reader, to confirm): Which of these provisions governs this recei
      ... -> rental_improvement
  rental_improvement(New roof)  [IRC 263(a); IRC 168(c); Rev. Proc. 87-57 Table A-6]
= 233.38

?- receipt_treatment("Groceries", 96.40)
receipt_treatment(Groceries)  [IRC 162; IRC 262; IRC 274(n); IRC 263(a); IRC 280A]
  assumed (AGENT receipt_reader, to confirm): Which of these provisions governs this recei
      ... -> personal_expense
  personal_expense(Groceries)  [IRC 262]
= 0.00
```

The agent's contribution is the choice of provision on each line marked *assumed*, and the amounts come from the rules. The $90.00 is the 50% limit of §274(n)(1) applied to a $180 dinner. The $233.38 is 1.667% of $14,000, which is the first-year rate in the 27.5-year table for residential rental property placed in service in July, and since a new roof is an improvement to the building, the law spreads

its cost over the building's 27.5-year life. The groceries are a personal expense under §262, and nothing is deducted.

6.3 A loop the law requires

Consider a self-employed taxpayer who buys health coverage on the marketplace. She has $58,000 of income before the health insurance deduction and pays $9,600 of premiums, the benchmark premium for her area is $8,400, and the poverty line for her household is $15,060.

```
RULE premium_tax_credit (magi) CITE "IRC 36B(b)"
  = MAX(FACT return.slcsp - TABLE("ptc_applicable_pct_2025", magi / FACT return.fpl) * magi, 0)
END

RULE sehi_and_ptc CITE "IRC 162(l); IRC 36B; Rev. Proc. 2014-41 sec. 4" {
  LET d = 0
  REPEAT {
    LET ptc = MIN(premium_tax_credit(FACT return.agi_before_sehi - d), FACT return.premiums)
    LET d_next = MIN(FACT return.premiums - ptc,
                  FACT return.schedule_c_net - FACT return.se_tax / 2)
    LET delta = d_next - d
    LET d = d_next
  } UNTIL delta < 1 AND delta > -1 LIMIT 25 END
  SET FORM "Schedule 1" LINE "17" = d
  SET FORM "8962" LINE "24" = ptc
} END
```

Each pass recomputes the credit from income net of the deduction, and then the deduction from premiums net of the credit:

```
?- sehi_and_ptc

pass   premium tax credit   SE health deduction
   1             3,685.67              5,914.33
   2             4,677.77              4,922.23
   3             4,519.46              5,080.54
   4             4,544.94              5,055.06
   5             4,540.84              5,059.16
   6             4,541.50              5,058.50

REPEAT converged after 6 passes
Schedule 1 line 17 = 5,058.50
Form 8962 line 24  = 4,541.50
```

The two values move toward each other from opposite sides, overshooting by less each time, and after six passes they agree to the dollar, with a credit of $4,541.50 and a deduction of $5,058.50. `LIMIT 25` guarantees that the loop would have ended even if they had not converged.

6.4 Scenarios, from rules to returns

We now return to the consultant from the introduction, who is single, lives in California, and has $180,000 of receipts and $30,000 of expenses on Schedule C. The facts given to the resolver are:

```
FACT taxpayer.married_at_year_end         = false
FACT taxpayer.paid_over_half_home_cost    = false
FACT taxpayer.dependents                  = []
FACT taxpayer.entity_type                 = "none"
FACT taxpayer.has_other_plan              = false
FACT taxpayer.occupation                  = "software consultant"
FACT taxpayer.metro_area                  = "San Jose"
```

```
FACT return.schedule_c_gross              = 180,000.00
FACT return.schedule_c_expenses           = 30,000.00
FACT return.schedule_c_net                = 150,000.00
FACT return.se_tax                        = 21,194.00
FACT return.magi                          = 139,403.00
FACT return.agi                           = 139,403.00
FACT return.filing_status                 = "SINGLE"
FACT return.mortgage_interest             = 0.00
FACT return.charitable_cash               = 2,000.00
FACT return.state_local_taxes             = 8,719.00
FACT return.medical_expenses              = 0.00
FACT plan.next_year_charitable            = 2,000.00
```

The rules that make up her alternatives are listed below. The S corporation option asks the wage agent for a market salary and uses at least 40% of profit. The SEP-IRA option allows 20% of net earnings after half of the self-employment tax. The solo 401(k) is offered only where the S corporation exists, since its contributions are computed from the corporate salary. California's S corporation tax is a rule of its own, because it depends on the K-1 after every option that changes the K-1. Finally, the bunching option moves next year's planned $2,000 of gifts into this year.

```
AGENT wage_researcher
  MODEL "language model with search"
  ROLE "Find published wage data (BLS OES) for an occupation and metro area."
  TOOLS (web_search)
END

RULE scorp_option CITE "IRC 1361; IRC 1362; IRC 1366; IRC 3111" {
  IF FACT return.schedule_c_net > 60000 AND FACT taxpayer.entity_type == "none" THEN {
    LET market = AGENT(wage_researcher, "Median annual wage for this occupation and metro area",
                   FACT taxpayer.occupation, FACT taxpayer.metro_area) EXPECT money CONFIRM
    LET salary = MAX(market, 40% * FACT return.schedule_c_net)
    LET employer_fica = 7.65% * salary
    LET k1 = FACT return.schedule_c_net - salary - employer_fica - 42 - 1200
    OPTION form_llc_elect_s "Form an LLC and elect S-corporation status (Form 2553)"
      COST employer_fica + 42 + 1200 {
      SET FORM "W-2" LINE "1" = salary
      SET FORM "Schedule K-1 (1120-S)" LINE "1" = k1
      SET FORM "1040" LINE "schedule_c" = 0
      SET FACT return.schedule_c_net = 0
      SET FACT return.k1_ordinary = k1
      SET FACT return.k1_final = k1
      SET FACT return.scorp_salary = salary
      SET FACT scenario.employee_fica = 7.65% * salary
      SET FACT taxpayer.entity_type = "s_corp"
      USE form_1120s
    }
  } END
} END

RULE sep_ira_option CITE "IRC 404(a)(8); IRC 402(h); IRC 415(c)" {
  IF FACT return.schedule_c_net > 0 AND NOT FACT taxpayer.has_other_plan == YES THEN {
    LET limit = MIN(20% * (FACT return.schedule_c_net - FACT return.se_tax / 2), 70000)
    OPTION open_sep_ira "Open a SEP-IRA and contribute before the filing deadline"
      DEFERS limit INTO YEARS 30 {
      SET FORM "Schedule 1" LINE "16" = limit
    }
  } END
} END

RULE solo_401k_option CITE "IRC 401(k); IRC 402(g); IRC 404(a)(3); IRC 415(c)" {
```

```
  IF FACT taxpayer.entity_type == "s_corp" AND NOT FACT taxpayer.has_other_plan == YES THEN {
    LET deferral = MIN(23500, FACT return.scorp_salary)
    LET employer = 25% * FACT return.scorp_salary
    OPTION open_solo_401k "Adopt a one-participant 401(k) in the S corporation"
      DEFERS deferral + employer INTO YEARS 30 {
      SET FORM "W-2" LINE "1" = FACT return.scorp_salary - deferral
      SET FORM "W-2" LINE "12D" = deferral
      SET FORM "Schedule K-1 (1120-S)" LINE "1" = FACT return.k1_ordinary - employer
      SET FACT return.k1_final = FACT return.k1_ordinary - employer
    }
  } END
} END

RULE ca_scorp_tax CITE "Cal. Rev. & Tax. Code 23802(b); 23153" {
  IF FACT taxpayer.entity_type == "s_corp" THEN {
    SET FACT scenario.entity_tax = MAX(800, 1.5% * FACT return.k1_final)
  } END
} END

RULE bunch_charity_option CITE "IRC 170; IRC 63" {
  IF FACT return.charitable_cash > 0 AND FACT plan.next_year_charitable > 0 THEN {
    OPTION bunch_gifts "Give next year's planned gifts this year (or through a donor-advised fund)
        "
      COST 0 {
      SET FORM "Schedule A" LINE "11" = FACT return.charitable_cash + FACT plan.
          next_year_charitable
      SET FACT return.charitable_cash = FACT return.charitable_cash + FACT plan.
          next_year_charitable
      SET FACT plan.next_year_charitable = 0
    }
  } END
} END

RULE deduction CITE "IRC 63(b); IRC 63(e)" {
  ELECT method
    CHOICE standard { SET FORM "1040" LINE "12e" = standard_deduction }
    CHOICE itemized { SET FORM "1040" LINE "12e" = itemized_total
                      USE schedule_a }
  END
} END

GUARD reasonable_compensation CITE "IRC 3121(a); Rev. Rul. 74-44"
  IF FACT scenario.salary < 40% * FACT return.schedule_c_net
  THEN REJECT "S-corporation salary below a defensible share of profit"
END
```

The resolver orders the entry rules by what they write and read, enumerates the branches, hands each one's form lines to the engine, and settles the deduction election inside each scenario:

```
stratified order: filing_status -> scorp_option -> sep_ira_option -> solo_401k_option ->
    ca_scorp_tax -> bunch_charity_option -> deduction
branches enumerated: 16; after settling the separable deduction election: 8
```

Each scenario appears twice in the table, once with the gifts bunched and once without, and the two rows agree. The reason is that $4,000 of gifts and $8,719 of state tax add up to well under the $15,750 standard deduction, so the engine took the standard deduction in every scenario and bunching made no difference. A taxpayer with a mortgage might well see a different result, and the model would find it in exactly the same way.

Scenario	Federal	CA	FICA	Costs	Total	Deferred
S-corp + solo 401(k) + standard	11,386	5,775	6,120	7,362	30,643	43,500
S-corp + solo 401(k) + bunched gifts + standard	11,386	5,775	6,120	7,362	30,643	43,500
Sch C + SEP-IRA + standard	32,966	6,127	0	0	39,092	27,881
Sch C + SEP-IRA + bunched gifts + standard	32,966	6,127	0	0	39,092	27,881
S-corp + standard	20,293	9,960	6,120	7,362	43,735	0
S-corp + bunched gifts + standard	20,293	9,960	6,120	7,362	43,735	0
Sch C + standard	37,872	8,719	0	0	46,591	0
Sch C + bunched gifts + standard	37,872	8,719	0	0	46,591	0

Table 1: The scenarios of Section 6.4 in dollars, tax year 2025, from the run above. Federal is income tax plus, for a sole proprietor, self-employment tax. CA includes California's S corporation tax. FICA is the employee share of payroll tax on the salary. Costs are the employer's payroll tax, FUTA and $1,200 of administration. Deferred is the retirement contribution on which tax is deferred, which is reported apart from the total.

The best scenario was produced by the following sequence of rules, choices and writes, and this trace is what the user sees beside the recommendation:

```
best: S-corp + solo 401(k) + standard

filing_status  [IRC 1; IRC 2; IRC 6013]
head_of_household_eligible()  [IRC 2(b)]
  SET FORM "1040" LINE "status" = SINGLE
scorp_option  [IRC 1361; IRC 1362; IRC 1366; IRC 3111]
assumed (AGENT wage_researcher, to confirm): Median annual wage for this occupation and m
    ... -> 80000.0
  LET market = 80,000.00
  LET salary = 80,000.00
  LET employer_fica = 6,120.00
  LET k1 = 62,638.00
  OPTION form_llc_elect_s TAKEN: Form an LLC and elect S-corporation status (Form 2553);
      COST 7,362.00
    SET FORM "W-2" LINE "1" = 80,000.00
    SET FORM "Schedule K-1 (1120-S)" LINE "1" = 62,638.00
    SET FORM "1040" LINE "schedule_c" = 0.00
    SET FACT return.schedule_c_net = 0.00
    SET FACT return.k1_ordinary = 62,638.00
    SET FACT return.k1_final = 62,638.00
    SET FACT return.scorp_salary = 80,000.00
    SET FACT scenario.employee_fica = 6,120.00
    SET FACT taxpayer.entity_type = s_corp
    USE form_1120s
sep_ira_option  [IRC 404(a)(8); IRC 402(h); IRC 415(c)]
  IF -> false: nothing offered in this scenario
solo_401k_option  [IRC 401(k); IRC 402(g); IRC 404(a)(3); IRC 415(c)]
  LET deferral = 23,500.00
  LET employer = 20,000.00
  OPTION open_solo_401k TAKEN: Adopt a one-participant 401(k) in the S corporation; COST
      0.00; DEFERS 43,500.00
    SET FORM "W-2" LINE "1" = 56,500.00
    SET FORM "W-2" LINE "12D" = 23,500.00
    SET FORM "Schedule K-1 (1120-S)" LINE "1" = 42,638.00
    SET FACT return.k1_final = 42,638.00
ca_scorp_tax  [Cal. Rev. & Tax. Code 23802(b); 23153]
  SET FACT scenario.entity_tax = 800.00
bunch_charity_option  [IRC 170; IRC 63]
  OPTION bunch_gifts not taken
deduction  [IRC 63(b); IRC 63(e)]
  ELECT method -> CHOICE standard
standard_deduction()  [IRC 63(c) as amended by P.L. 119-21]
```

```
    SET FORM "1040" LINE "12e" = 15,750.00
```

The engine received exactly these form lines:

```
best: S-corp + solo 401(k) + standard
choices: {"bunch_gifts": "skip", "form_llc_elect_s": "take", "method": "standard", "
    open_solo_401k": "take"}

SET FORM 1040 status                          = SINGLE
SET FORM W-2 1                                = 56,500.00
SET FORM Schedule K-1 (1120-S) 1              = 42,638.00
SET FORM 1040 schedule_c                      = 0.00
SET FORM W-2 12D                              = 23,500.00
SET FORM 1040 12e                             = 15,750.00
USE form_1120s
```

It is worth looking at where the savings come from. The S corporation alone saves $2,856 against the sole proprietorship, and three effects pull against one another to produce that figure. The payroll tax on an $80,000 salary, $6,120 from each side, replaces most of the $21,194 of self-employment tax; the §199A deduction is roughly halved, because only the K-1 income qualifies and the salary does not; and California charges the S corporation 1.5% of its income, with an $800 minimum. The SEP-IRA saves $7,499 this year, but $27,881 of the contribution behind that saving will be taxed when it is withdrawn. The S corporation with a solo 401(k) is the lowest at $30,643, because the salary supports a $23,500 employee deferral and a $20,000 employer contribution, and the $43,500 is likewise deferred. A multi-year objective with a discount rate and an assumed retirement tax rate orders these scenarios fairly, whereas a one-year objective overstates the retirement options, and so the model reports both numbers and lets the user see the difference.

Which assumptions matter. The $80,000 salary is the wage agent's answer and is marked `CONFIRM`, which raises the question of whether the user should actually be asked about it. To find out, we re-evaluated the S corporation scenario across a range of salaries with the same engine. At $60,000, $80,000 and $100,000 its totals are $40,474, $43,735 and $47,180, against $46,591 for the sole proprietorship, so the S corporation wins only for salaries below about $96,700. If the plausible range of salaries for her occupation reaches $96,700, the salary decides the recommendation and the user is asked to confirm it; if the range stays well below that point, the assumption is reported with its margin and the user is not troubled with it. This is how the model decides which of its assumptions are worth a question.

6.5 The same machinery in other domains

Nothing in the machinery above refers to tax. To show this, we wrote three small rule packs from other fields and ran them through the same resolver.

Medicine. An ECG agent answers with the name of a rhythm rule, and the rhythm rule applies a published risk score:

```
RULE rhythm (pt) CITE "2023 ACC/AHA/ACCP/HRS AF guideline"
  = AGENT(ecg_reader, "Which rhythm is this?", FACT pt.ecg)
    EXPECT RULE IN (atrial_fibrillation, atrial_flutter, sinus_rhythm) CONFIRM
END

RULE cha2ds2_vasc (pt) CITE "Lip et al., Chest 2010"
  = FACT pt.chf + FACT pt.hypertension + FACT pt.diabetes + FACT pt.vascular_disease
    + 2 * FACT pt.prior_stroke_tia + FACT pt.female
    + MAX(MIN(FACT pt.age - 64, 1), 0) + MAX(MIN(FACT pt.age - 74, 1), 0)
END
```

```
RULE anticoagulation_indicated (pt) CITE "2023 ACC/AHA/ACCP/HRS AF guideline"
  = (FACT pt.female == NO AND cha2ds2_vasc(pt) >= 2)
    OR (FACT pt.female == YES AND cha2ds2_vasc(pt) >= 3)
END

RULE atrial_fibrillation (pt) CITE "2023 ACC/AHA/ACCP/HRS AF guideline" {
  IF anticoagulation_indicated(pt) THEN {
    ELECT agent_choice
      CHOICE doac { SET FORM "order" LINE "anticoagulant" = "DOAC" }
      CHOICE warfarin { SET FORM "order" LINE "anticoagulant" = "warfarin, INR 2-3" }
    END
  } END
} END

GUARD mechanical_valve CITE "DOACs contraindicated with mechanical heart valves"
  IF FACT scenario.anticoagulant == "DOAC" AND FACT patient.mechanical_valve == YES
  THEN REJECT "DOAC contraindicated with a mechanical valve"
END
```

Biology. Each couplet of an identification key asks an image agent one question, and the answer names either the next couplet or a species:

```
AGENT leaf_reader
  MODEL "vision model"
  ROLE "Answer one couplet of a botanical key from a photograph of a leaf."
  TOOLS (image)
END

RULE couplet_1 (s) CITE "illustrative key to common oaks"
  = AGENT(leaf_reader, "Are the leaf lobes pointed with bristle tips, or rounded?", FACT s.photo)
    EXPECT RULE IN (couplet_2, couplet_3)
END

RULE couplet_3 (s) CITE "white oak group"
  = AGENT(leaf_reader, "Is the leaf underside pale and densely hairy?", FACT s.photo)
    EXPECT RULE IN (quercus_bicolor, quercus_alba)
END

RULE quercus_bicolor (s) CITE "swamp white oak" = "Quercus bicolor" END
```

Law. California's cap on residential security deposits after AB 12:

```
RULE small_landlord (lease) CITE "Cal. Civ. Code 1950.5(c)(4), AB 12 (2023), effective July 1,
    2024"
  = FACT lease.landlord_is_natural_person_or_family_llc == YES
    AND FACT lease.landlord_rental_properties <= 2 AND FACT lease.landlord_units <= 4
END

RULE deposit_cap (lease) CITE "Cal. Civ. Code 1950.5(c), AB 12 (2023)" {
  IF small_landlord(lease) THEN { LET result = 2 * FACT lease.monthly_rent }
  ELSE { LET result = FACT lease.monthly_rent } END
} END
```

Resolving each of the three gives:

```
?- rhythm(patient)
rhythm(patient)  [2023 ACC/AHA/ACCP/HRS AF guideline]
  assumed (AGENT ecg_reader, to confirm): Which rhythm is this?... -> atrial_fibrillation
  atrial_fibrillation(patient)  [2023 ACC/AHA/ACCP/HRS AF guideline]
    anticoagulation_indicated(patient)  [2023 ACC/AHA/ACCP/HRS AF guideline]
```

```
      cha2ds2_vasc(patient)  [Lip et al., Chest 2010]
   cha2ds2_vasc(patient) = 3; anticoagulation_indicated = YES
   ELECT agent_choice: doac | warfarin  (each branch is a scenario; GUARD mechanical_valve
        applies)

?- couplet_1(leaf)
couplet_1(leaf)  [illustrative key to common oaks]
  assumed (AGENT leaf_reader): Are the leaf lobes pointed with bristle tips... ->
      couplet_3
  couplet_3(leaf)  [white oak group]
    assumed (AGENT leaf_reader): Is the leaf underside pale and densely hairy... ->
       quercus_bicolor
    quercus_bicolor(leaf)  [swamp white oak]
= Quercus bicolor

?- deposit_cap(lease)
deposit_cap(lease)  [Cal. Civ. Code 1950.5(c), AB 12 (2023)]
  small_landlord(lease)  [Cal. Civ. Code 1950.5(c)(4), AB 12 (2023), effective July 1,
      2024]
= 6,000.00
```

7 Beyond tax

Nothing in the language is specific to tax except the rule packs. Any body of rules that decides outcomes from facts, allows choices and admits actions with costs can be written in the same way, and the resolver, the agents, the enumerator and the guards carry over unchanged, with `SET FORM` writing a field of whatever document the domain produces, be it an order, a report or a notice.

In *medicine*, clinical guidelines, drug-interaction rules and coverage criteria are rule packs; the options are tests and treatments with their costs and risks, the guards are contraindications, and the objective may combine the expected outcome with the patient's cost under the patient's own coverage. In *biology*, an identification key is the purest case of an answer that is a rule, and in strain design, gene deletions are options with costs, a viability guard rejects designs that are predicted not to grow, and the objective trades yield against the number of edits. In *law*, statutes, regulations and contracts are rule packs, forum, settlement and procedural choices are options with costs and deadlines, and limitations periods are guards; benefits eligibility, immigration categories and export classification follow the same pattern. Section 6.5 shows one example of each of these running. The same holds for building codes, degree requirements, lending and insurance rules, and the analysis of proposed legislation, in which one rule pack is evaluated across a whole population of fact sets.

8 Limits

We would like to be plain about what the model cannot do. It is only as complete as its rule packs, so a provision that has not been written is a scenario that will not be found, and writing the rules for one year is a large body of work, although most of each year inherits from the year before. The enumeration is exponential in the number of interacting options; for an individual, ordering and local settlement keep it small, as we saw in Section 6.4, but a business with many entities will need the search methods of the corporate restructuring literature.

Agent answers are assumptions, and a wrong answer leads to a wrong scenario. Confining the answers to listed rules, recording their provenance and asking the user to confirm the ones that matter all reduce this risk, but they cannot remove it entirely. The guards encode the common anti-abuse doctrines, though not all of them, and the reasonable salary in our example is an input

that the taxpayer must be able to defend to the IRS. For all of these reasons, the model makes recommendations, and the taxpayer and the preparer remain responsible for the return that is filed.

9 Conclusion

In this paper, we have described a model in which each provision of the tax law is a named rule and a name can stand for a rule wherever a value can. This one idea brings three problems together. The interview becomes the set of facts that resolution reaches; the use of a language model is confined to choosing among named rules and supplying assumed facts, each of which is recorded and, where it matters, confirmed by the user; and planning becomes the enumeration of the elections and options that the rules contain, with every scenario computed into the same fixed set of forms by the same engine that files the return. The language grew out of a command language for a desktop assistant, and it keeps that language's central idea, which is that a name is a value, so that programs are built from programs.

References

[1] M. J. Sergot, F. Sadri, R. A. Kowalski, F. Kriwaczek, P. Hammond, H. T. Cory, *The British Nationality Act as a logic program*, Communications of the ACM 29(5), 1986.

[2] D. M. Sherman, *A Prolog model of the Income Tax Act of Canada*, Proc. ICAIL 1987.

[3] S. B. Lawsky, *A logic for statutes*, Florida Tax Review 21, 2017.

[4] D. Merigoux, N. Chataing, J. Protzenko, *Catala: a programming language for the law*, Proc. ACM Program. Lang. 5 (ICFP), 2021.

[5] Internal Revenue Service, *Fact Graph*, github.com/IRS-Public/fact-graph, 2025.

[6] Intuit Inc., patents built on its tax calculation and completeness graphs, e.g. U.S. Pat. Nos. 11,222,384 and 11,386,505 (see their descriptions of calculation and completeness graphs).

[7] U.S. Pat. No. 7,970,683, *System and method for multi-state tax analysis*.

[8] M. J. Sergot, *A query-the-user facility for logic programming*, in Integrated Interactive Computing Systems, North-Holland, 1983.

[9] J. Arias, M. Moreno-Rebato, J. A. Rodríguez-García, S. Ossowski, *Automated legal reasoning with discretion to act using s(LAW)*, arXiv:2401.14511, 2024.

[10] L. Pan, A. Albalak, X. Wang, W. Y. Wang, *Logic-LM: empowering large language models with symbolic solvers for faithful logical reasoning*, Findings of EMNLP 2023.

[11] Leahy–Smith America Invents Act, Pub. L. 112-29, §14 (2011).

[12] Rev. Proc. 2014-41, 2014-33 I.R.B. 364.