Models for Metamath

Mario Carneiro

Carnegie Mellon University, Pittsburgh PA, USA

Abstract. Although some work has been done on the metamathematics of Metamath, there has not been a clear definition of a model for a Metamath formal system. We define the collection of models of an arbitrary Metamath formal system, both for tree-based and string-based representations. This definition is demonstrated with examples for propositional calculus, ZFC set theory with classes, and Hofstadter's MIU system, with applications for proving that statements are not provable, showing consistency of the main Metamath database (assuming ZFC has a model), developing new independence proofs, and proving a form of Gödel's completeness theorem.

Keywords: Metamath · Model theory · formal proof · consistency · ZFC · Mathematical logic

1 Introduction

Metamath is a proof language, developed in 1992, on the principle of minimizing the foundational logic to as little as possible [1]. An expression in Metamath is a string of constants and variables headed by a constant called the expression's "typecode". The variables are typed and can be substituted for expressions with the same typecode. See § 2.1 for a precise definition of a formal system, which mirrors the specification of the `.mm` file format itself.

The logic on which Metamath is based was originally defined by Tarski in [2]. Notably, this involves a notion of "direct" or "non-capturing" substitution, which means that no α-renaming occurs during a substitution for a variable. Instead, this is replaced by a "distinct variable" condition saying that certain substitutions are not valid if they contain a certain variable (regardless of whether the variable is free or not—Metamath doesn't know what a free variable is). For instance, the expression $\forall x\, \varphi$ contains a variable φ inside a binding expression "$\forall x\, \square$". (Metamath also does not have a concept of "binding expression", but it is safe to say that under a usual interpretation this would be considered a binding expression.) If there is a distinct variable condition between x and φ, then the substitution $\varphi \mapsto x = y$ is invalid, because x is present in the substitution to φ. This is stricter than the usual first-order logic statement "x is not free in φ", because $\varphi \mapsto \forall x\, x = y$ is also invalid. If there is no such distinct variable condition between x and φ, these substitutions would be allowed, and applying them to $\forall x\, \varphi$ would result in $\forall x\, x = y$ and $\forall x \forall x\, x = y$, respectively.

In this paper, we will develop a definition for models of Metamath-style formal systems, which will operate by associating a function to each syntactical

construct according to its type. For example, the forall symbol is defined by the axiom "wff $\forall x\, \varphi$", which is to say it takes as input a set variable and a wff variable, and produces a wff expression. This construct is associated to an interpretation function $\pi_\forall : \mathsf{U}_{\text{set}} \times \mathsf{U}_{\text{wff}} \to \mathsf{U}_{\text{wff}}$, where U_{set} is the universe of set variables and U_{wff} is the universe of wff variables, which are each provided as part of the definition of a model.

Note the difference from the usual signature of the forall, $\pi'_\forall : (M \to \mathsf{Bool}) \to \mathsf{Bool}$, which maps functions from the model universe M to boolean values, to a boolean value. In order to make our definition work, we need the set U_{wff} to be more complicated than just Bool. Instead, it is effectively $(V \to M) \to \mathsf{Bool}$, that is, a function from assignments of variables to elements of the model, to a boolean value. In other words, a wff can be thought of as an infinite-place predicate $\varphi(v_0, v_1, v_2, \dots)$ (although the value can only depend on finitely many of the provided variables).

1.1 Grammars and trees

Unfortunately, although it is possible to define what it means to be a model for any Metamath formal system, we can't quite reduce it to a collection of interpretation functions, like it is normally done, without a way to parse the strings which are used in the proof. This leads to the idea of grammatical parsing, which we take up in earnest in § 2.3. By separating all axioms into "syntax axioms" and "logical axioms", we can find an isomorphism to a representation of statements as trees, with syntax axioms forming the nodes of the tree. Most interesting Metamath systems are grammatical, but for example Hofstadter's MIU system [3], formalized in Metamath as `miu.mm`, is a valid formal system which is not grammatical (see § 3.2).

The main work is presented in § 2. A short recap of Metamath's formalism as it will be used in this work is in § 2.1, followed by the definition of a model in § 2.2. Then we define the subset of "grammatical" formal systems, which are those for which parsing is possible, in § 2.3, and rebuild the theory for a tree representation of formal systems in § 2.4. The model theory of tree formal systems is developed in § 2.5. A selection of examples is provided in § 3, and in particular we prove that Metamath's ZFC formalization, `set.mm`, has a model in Theorem 4. Some applications of model theory are developed in § 4, finishing with a proof of Gödel's completeness theorem in § 4.2.

2 Formal definition

2.1 Metamath recap

We recall the definitions from Appendix C of the Metamath book [1], but with a slight modification for a global type function.

1. Let CN, VR be disjoint sets, called the set of *constants* and *variables* respectively.

CN	constants	§ 2.1	VR	variables	§ 2.1	Type	type of expr	§ 2.1
EX	expressions	§ 2.1	DV	distinct variables	§ 2.1	$\mathcal{V}$	variables in expr	§ 2.1
σ	substitution	§ 2.1	VH	variable hypotheses	§ 2.1	TC	typecodes	§ 2.1
VT	variable typecodes	§ 2.1	U	universe	§ 2.2	VL, μ	valuations	§ 2.2
#	freshness relation	§ 2.2	η	interpretation	§ 2.2	SA	syntax axioms	§ 2.3
Syn	syntax for expr	§ 2.4	ST	syntax trees	§ 2.4	π	interpretation (tree)	§ 2.5

Table 1. Definition cheat sheet

2. Let Type : $\mathsf{VR} \to \mathsf{CN}$ be a function, understood to map a variable to its typecode constant.
3. Let $\mathsf{VT} = \{\text{Type}(v) \mid v \in \mathsf{VR}\}$ be the set of typecodes of variables.
4. $\mathsf{EX} = \{e \in \bigcup_{n\in\omega}{}^{n}(\mathsf{CN} \cup \mathsf{VR}) \mid (|e| > 0 \wedge e_0 \in \mathsf{CN})\}$ (the set of expressions),
5. $\mathsf{DV} = \{x \subseteq \mathsf{VR} \mid |x| = 2\}$ (the set of distinct variable specifications), and
6. $\mathcal{V}(e) = \mathsf{VR} \cap \{e_n \mid 0 \le n < |e|\}$ (the set of variables in an expression).
7. We also write $\text{Type}(e) = e_0$ for $e \in \mathsf{EX}$.
8. A *substitution* is a function $\sigma : \mathsf{EX} \to \mathsf{EX}$ such that $\sigma(\langle c \rangle) = \langle c \rangle$ for $c \in \mathsf{CN}$ and $\sigma(gh) = \sigma(g)\sigma(h)$, where adjacency denotes concatenation of sequences. (Such a function is determined by its values on $\{\langle v \rangle \mid v \in \mathsf{VR}\}$.)
9. Define $\mathsf{VH}_v = \langle \text{Type}(v), v \rangle$, for $v \in \mathsf{VR}$ (a *variable hypothesis*).
10. A *pre-statement* is a tuple $\langle D, H, A \rangle$ where $D \subseteq \mathsf{DV}$, $H \subseteq \mathsf{EX}$ is finite, and $A \in \mathsf{EX}$.
11. The *reduct* of $\langle D, H, A \rangle$ is $\langle D_M, H, A \rangle$ where $D_M = D \cap \mathscr{P}(\mathcal{V}(H \cup \{A\}))$, and a statement is defined as the reduct of some pre-statement.
12. A *formal system* is a tuple $\langle \mathsf{CN}, \mathsf{VR}, \text{Type}, \Gamma \rangle$ where $\mathsf{CN}, \mathsf{VR}, \text{Type}$ are as above and Γ is a set of statements.
13. The *closure* of a set $H \subseteq \mathsf{EX}$ relative to D is the smallest set C such that:
    - $H \cup \{\mathsf{VH}_v \mid v \in \mathsf{VR}\} \subseteq C$
    - For every $\langle D', H', A' \rangle \in \Gamma$ and every substitution σ, if
      - For all $e \in H' \cup \{\mathsf{VH}_v \mid v \in \mathsf{VR}\}$, $\sigma(e) \in C$, and
      - For all $\{\alpha, \beta\} \in D'$, if $\gamma \in \mathcal{V}(\sigma(\mathsf{VH}_\alpha))$ and $\delta \in \mathcal{V}(\sigma(\mathsf{VH}_\beta))$, then $\{\gamma, \delta\} \in D$,

      then $\sigma(A') \in C$.
14. A pre-statement $\langle D, H, A \rangle$ is *provable* if A is in the closure of H relative to D, and a theorem is a statement that is the reduct of a provable pre-statement.
15. Let TC be the set of typecodes of theorems. (Explicitly, this is $\mathsf{TC} = \mathsf{VT} \cup \{\text{Type}(A) \mid \langle D, H, A \rangle \in \Gamma\}$.)
16. Two formal systems $\langle \mathsf{CN}, \mathsf{VR}, \text{Type}, \Gamma \rangle$ and $\langle \mathsf{CN}, \mathsf{VR}, \text{Type}, \Gamma' \rangle$ are *equivalent* if they generate the same set of theorems (or equivalently, if every axiom in one is a theorem of the other).

Why a global type function? A careful comparison with Appendix C of the Metamath book [1] shows that in the original definition a variable only has a type locally (inside a statement), while we require all variables to have a unique and globally defined type, provided by the Type function. In practice, variables are never reintroduced with a different type, so this is not a strong requirement.

Additionally, there is some ongoing work to amend the specification to disallow such multi-typed variables.

Nevertheless, it is a simple fix to convert a formal system with multi-typed variables to one with a global type function: Take the set of variables to be $\mathsf{VT} \times \mathsf{VR}$, and define $\mathrm{Type}(c, v) = c$. Then whenever a variable v appears in a statement with type c, use the variable $\langle c, v \rangle$ instead. This is equivalent to just prepending the type of the variable to its name, so that uses of the same variable with a different type are distinguished.

2.2 Models of formal systems

Fix a collection of sets U_c for $c \in \mathsf{TC}$, which will represent the "universe" of objects of each typecode.

Definition 1. *A* valuation *is a function μ on VR such that $\mu(v) \in \mathsf{U}_{\mathrm{Type}(v)}$ for all $v \in \mathsf{VR}$. The set of all valuations is denoted by VL.*

Definition 2. *A* freshness relation $\#$ *is a symmetric relation on the disjoint union $\bigsqcup \mathsf{U} = \bigsqcup_{c \in \mathsf{TC}} \mathsf{U}_c$ such that for any $c \in \mathsf{VT}$ and any finite set $W \subseteq \bigsqcup \mathsf{U}$, there is a $v \in \mathsf{U}_c$ with $v \# w$ for all $w \in W$.*

Definition 3. *A* model *of the formal system $\langle \mathsf{CN}, \mathsf{VR}, \mathrm{Type}, \Gamma \rangle$ is a tuple $\langle \mathsf{U}, \#, \eta \rangle$ where U is a function on TC and $\#$ is a freshness relation, and for each $\mu \in \mathsf{VL}$, η_μ is a partial function on EX such that:*

- *(Type correctness) For all $e \in \mathsf{EX}$, if $\eta_\mu(e)$ is defined then $\eta_\mu(e) \in \mathsf{U}_{\mathrm{Type}(e)}$.*
- *(Variable application) For all $v \in \mathsf{VR}$, $\eta_\mu(\mathsf{VH}_v) = \mu(v)$.*
- *(Axiom application) For each $\langle D, H, A \rangle \in \Gamma$, if*
  - *$\mu(\alpha) \# \mu(\beta)$ for all $\{\alpha, \beta\} \in D$, and*
  - *$\eta_\mu(h)$ is defined for all $h \in H$,*

  then $\eta_\mu(A)$ is defined.
- *(Substitution property) For each substitution σ and $e \in \mathsf{EX}$, $\eta_\mu(\sigma(e)) = \eta_{\sigma(\mu)}(e)$, where $\sigma(\mu) \in \mathsf{VL}$ is defined by $\sigma(\mu)(v) = \eta_\mu(\sigma(\mathsf{VH}_v))$.*
- *(Dependence on present variables) For all $\nu \in \mathsf{VL}$, $e \in \mathsf{EX}$, If $\mu(v) = \nu(v)$ for all $v \in \mathcal{V}(e)$, then $\eta_\mu(e) = \eta_\nu(e)$.*
- *(Freshness substitution) For all $v \in \bigsqcup \mathsf{U}$, $e \in \mathsf{EX}$, if $\eta_\mu(e)$ is defined and $v \# \mu(w)$ for all $w \in \mathcal{V}(e)$, then $v \# \eta_\mu(e)$.*

Here equality means that one side is defined iff the other is and they have the same value. We say that $e \in \mathsf{EX}$ is true in the model *if $\eta_\mu(e)$ is defined for all $\mu \in \mathsf{VL}$.*

The key property of a model is *soundness*, the fact that the axiom application law applies also to theorems.

Theorem 1. *For any theorem $\langle D, H, A \rangle$, if $\mu(\alpha) \# \mu(\beta)$ for all $\{\alpha, \beta\} \in D$ and $\eta_\mu(h)$ is defined for all $h \in H$, then $\eta_\mu(A)$ is defined.*

Proof. By dependence on present variables, we may replace μ by any other μ' such that $\mu(v) = \mu'(v)$ for all $v \in \mathcal{V}(H \cup \{A\})$ without affecting the truth of the hypotheses or conclusion. If $\langle D, H, A\rangle$ is the reduct of $\langle D', H, A\rangle$ where D' refers to finitely many additional variables (i.e. $D' \subseteq \mathcal{P}(V)$ for some finite set $V \supseteq \mathcal{V}(H \cup \{A\})$), order these as $V = \{v_1, \ldots, v_n\}$ with $\mathcal{V}(H \cup \{A\}) = \{v_1, \ldots, v_k\}$ for some $k \leq n$. Then use the freshness constraint to recursively select values $\mu'(v_i)$ for each $k < i \leq n$ such that $\mu'(v_i) \# \mu'(v_j)$ for all $j < i$. Then this new μ' will satisfy the hypothesis $\mu'(\alpha) \# \mu'(\beta)$ for all $\{\alpha, \beta\} \in D'$, so that it suffices to prove the theorem for provable pre-statements.

We prove by induction that whenever A is in the closure of H relative to D, $\eta_\mu(A)$ is defined. If $A \in H$, it is true by assumption, and if $A = \mathsf{VH}_v$ for some $v \in \mathsf{VR}$, then it is true by the variable application law. Otherwise we are given $\langle D', H', A'\rangle \in \Gamma$ and a substitution σ, such that for all $e \in H' \cup \{\mathsf{VH}_v \mid v \in \mathsf{VR}\}$, $\eta_\mu(\sigma(e))$ is defined (by the induction hypothesis), and for all $\{\alpha, \beta\} \in D'$, if $\gamma \in \mathcal{V}(\sigma(\mathsf{VH}_\alpha))$ and $\delta \in \mathcal{V}(\sigma(\mathsf{VH}_\beta))$, then $\{\gamma, \delta\} \in D$, and we wish to show that $\eta_\mu(A)$, with $A = \sigma(A')$, is defined.

For each $\gamma \in \mathcal{V}(\sigma(\mathsf{VH}_\alpha))$ and $\delta \in \mathcal{V}(\sigma(\mathsf{VH}_\beta))$, $\{\gamma, \delta\} \in D$ implies $\mu(\gamma) \# \mu(\delta)$ from the theorem hypothesis, hence by freshness substitution on the left and the right, $\mu(\gamma) \# \eta_\mu(\sigma(\mathsf{VH}_\beta))$, and then $\eta_\mu(\sigma(\mathsf{VH}_\alpha)) \# \eta_\mu(\sigma(\mathsf{VH}_\beta))$, or equivalently, $\sigma(\mu)(\alpha) \# \sigma(\mu)(\beta)$ for each $\{\alpha, \beta\} \in D'$.

Apply the axiom application law to $\sigma(\mu)$ and $\langle D', H', A'\rangle$. The substitution property reduces $\eta_\mu(\sigma(e))$ to $\eta_{\sigma(\mu)}(e)$ in the hypotheses, and $\eta_{\sigma(\mu)}(A')$ to $\eta_\mu(\sigma(A'))$ in the conclusion, hence $\eta_\mu(\sigma(A'))$ is defined, as we wished to show. □

In particular, if $\langle \emptyset, \emptyset, A\rangle$ is provable, then $\eta_\mu(A)$ is defined for all $\mu \in \mathsf{VL}$, which makes it a useful technique for showing that certain strings are not provable (see § 4.1).

For any formal system, there is a model, where $\mathsf{U}_c = \{*\}$ for each c, $* \# *$ is true, and $\eta_\mu(e) = *$ for all μ, e. Thus statements like "formal system X has a model" are not as useful here as they are in first-order logic. To marginalize this kind of model, we will call a model where each η_μ is a total function *trivial*. (We will have a slightly wider definition of trivial model given grammatical information, cf. Definition 6.)

Although this defines the property of being a model under the full generality of Metamath formal systems, the process simplifies considerably when expressions can be parsed according to a grammar.

2.3 Grammatical parsing

Definition 4. *A formal system is said to be* weakly grammatical *if for every $\langle D, H, A\rangle \in \Gamma$, if* $\mathrm{Type}(A) \in \mathsf{VT}$*, then there is some axiom $\langle \emptyset, \emptyset, A'\rangle \in \Gamma$ such that $\sigma(A') = A$ for some substitution σ and no variable occurs more than once in A'.*

For these systems we will define

$$\mathsf{SA} = \{A \mid \langle \emptyset, \emptyset, A\rangle \in \Gamma \wedge \mathrm{Type}(A) \in \mathsf{VT} \wedge \forall mn, (A_m = A_n \in \mathsf{VR} \rightarrow m = n)\},$$

the set of syntax axioms. *(We will identify the expression A with its statement $\langle \emptyset, \emptyset, A \rangle$ when discussing syntax axioms.)*

For example, any context-free grammar is a weakly grammatical formal system, where each production translates to a syntax axiom, and each nonterminal translates to a variable typecode. Most recursive definitions of a well-formed formula will fit this bill, although we can't capture the notion of bound variables with this alone.

Conversely, a weakly grammatical formal system yields a context-free grammar, where the terminals are $\mathsf{CN} \setminus \mathsf{VT}$, the non-terminals are VT, and for each $A \in \mathsf{SA}$ there is a production $\mathrm{Type}(A) \to \alpha$, where $\alpha_n = A_{n+1} \in \mathsf{CN}$ if $A_{n+1} \in \mathsf{CN}$ or $\alpha_n = \mathrm{Type}(A_{n+1}) \in \mathsf{VT}$ if $A_{n+1} \in \mathsf{VR}$. (This assumes that $A_{n+1} \in \mathsf{VT}$ is always false, but the two sets can be disjointified if this is not the case.)

Definition 5. *A* grammatical formal system *is a weakly grammatical formal system augmented with a function* $\mathrm{Syn} : \mathsf{TC} \to \mathsf{VT}$ *such that* $\mathrm{Syn}(c) = c$ *for all* $c \in \mathsf{VT}$ *and, defining* $\mathrm{Syn}(e)$ *for* $e \in \mathsf{EX}$ *such that* $\mathrm{Syn}(e)_0 = \mathrm{Syn}(e_0)$ *and* $\mathrm{Syn}(e)_n = e_n$ *for* $n > 0$, $\langle \emptyset, \emptyset, \mathrm{Syn}(e) \rangle$ *is a provable statement for all* $\langle D, H, A \rangle \in \Gamma$ *and* $e \in H \cup \{A\}$.

Remark 1. Of course, this notion is of interest primarily because it is satisfied by all major Metamath databases; in particular, `set.mm` is a grammatical formal system, with $\mathsf{VT} = \{\mathrm{set}, \mathrm{class}, \mathrm{wff}\}$, $\mathsf{TC} = \mathsf{VT} \cup \{\vdash\}$, and $\mathrm{Syn}(\vdash) = \mathrm{wff}$.

Definition 6. *A* model *of a grammatical formal system is a model in the sense of Definition 3 which additionally satisfies* $\mathsf{U}_c \subseteq \mathsf{U}_{\mathrm{Syn}(c)}$, $v \# w_c \leftrightarrow v \# w_{\mathrm{Syn}(c)}$ *when* $w \in \mathsf{U}_c$ *(and* w_c, $w_{\mathrm{Syn}(c)}$ *are its copies in the disjoint union), and* $\eta_\mu(e) = \eta_\mu(\mathrm{Syn}(e))$ *if the latter is in* U_c*, otherwise undefined. Such a model is* trivial *if* $\mathsf{U}_c = \mathsf{U}_{\mathrm{Syn}(c)}$ *for all* $c \in \mathsf{TC}$.

2.4 Tree representation of formal systems

The inductive definition of the closure of a set of statements immediately leads to a tree representation of proofs. A proof tree is a tree with nodes labeled by statements and edges labeled by expressions.

Definition 7. *We inductively define the statement "T is a* proof tree *for A" (relative to D, H) as follows:*

- *For each* $e \in H \cup \{\mathsf{VH}_v \mid v \in \mathsf{VR}\}$*, the single-node tree labeled by the reduct of* $\langle D, H, e \rangle$ *is a proof tree for* e.
- *For every* $\langle D', H', A' \rangle \in \Gamma$ *and every substitution* σ *satisfying the conditions for* $\sigma(A') \in C$ *in § 2.1.13, the tree labeled by* $\langle D', H', A' \rangle$ *with edges for each* $e \in H' \cup \mathcal{V}(H \cup \{A\})$ *leading to a proof tree for* $\sigma(e)$*, is a proof tree for* $\sigma(A')$.

The definition of closure ensures that there is a proof tree for A relative to D, H iff $\langle D, H, A\rangle$ is provable pre-statement. (The branches for variables outside $\mathcal{V}(H \cup \{A\})$ are discarded because they can always be replaced by the trivial substitution $\sigma(\langle v\rangle) = \langle v\rangle$ without affecting the closure deduction.) Additionally, we can prove by induction that every proof tree T encodes a unique expression $\mathrm{Expr}(T)$.

Definition 8. *An* unambiguous formal system *is a grammatical formal system whose associated context-free grammar is unambiguous.*

Remark 2. Note that for this to make sense we need SA to contain only axioms and not theorems, i.e. this property is not preserved by equivalence of formal systems. For such systems every Expr is an injection when restricted to the set ST of *syntax trees*, trees T relative to $\emptyset, \emptyset$ such that $\mathrm{Type}(T) := \mathrm{Type}(\mathrm{Expr}(T)) \in \mathsf{VT}$ (or equivalently, $\mathrm{Type}(T) = \mathrm{Type}(A) \in \mathsf{VT}$ where $\langle D, H, A\rangle$ is the root of T). The subtrees of a syntax tree are also syntax trees, and the nodes are syntax axioms, with variables (of the form VH_v) at the leaves.

With this, we can "rebuild" the whole theory using trees instead of strings, because the all valid substitutions have unique proof tree representations. The expressions in this new language will be trees, whose nodes are syntax axioms such as `wa` = "wff $(\varphi \wedge \psi)$", representing the "and" function, with variables at the leaves. However, we no longer need to know that `wa` has any structure of its own, besides the fact that it takes two wff variables and produces a wff. Thus we can discard the set CN entirely. (That is, the constant "(" has no meaning of its own here.)

Instead, we take as inputs to the construction the set TC' of typecodes, a set VR' of variables, a set SA' of things we call syntax axioms, although they have no internal structure, the function $\mathrm{Type}' : \mathsf{VR}' \cup \mathsf{SA}' \to \mathsf{VT}'$, as well as $\mathrm{Syn}' : \mathsf{TC}' \to \mathsf{VT}'$. A tree $T \in \mathsf{ST}'$ is either a variable from VR' or a syntax axiom $a \in \mathsf{SA}'$ connecting to more subtrees; each syntax axiom has a set v_i^a of variables labeling the edges, and a type $\mathrm{Type}'(a)$; $\mathrm{Type}'(T)$ is defined as the type of the root of T.

We replace EX in the string representation with EX', which consists of tuples $\langle c, T\rangle$ where $c \in \mathsf{TC}'$, $T \in \mathsf{ST}'$, and $\mathrm{Syn}'(c) = \mathrm{Type}'(T)$. We extend Type' to EX' by $\mathrm{Type}'(\langle c, T\rangle) = c$. $\mathcal{V}'(T)$ is defined by induction such that $\mathcal{V}'(v) = \{v\}$ and $\mathcal{V}'(T)$ at a syntax axiom is the union of $\mathcal{V}'(T_i)$ over the child subtrees T_i. A substitution σ is a function $\mathsf{ST}' \to \mathsf{ST}'$ such that $\sigma(a[T_1, \ldots, T_n]) = a[\sigma(T_1), \ldots, \sigma(T_n)]$ for each syntax axiom a, with the value at variables left undetermined, extended to $\mathsf{EX}' \to \mathsf{EX}'$ by $\sigma(\langle c, T\rangle) = \langle c, \sigma(T)\rangle$.

Pre-statements and statements are defined exactly as before: A pre-statement is a tuple $\langle D, H, A\rangle$ where $D \subseteq \mathsf{DV}'$, $H \subseteq \mathsf{EX}'$ is finite, and $A \in \mathsf{EX}'$. The reduct of $\langle D, H, A\rangle$ is $\langle D_M, H, A\rangle$ where $D_M = D \cap \mathscr{P}(\mathcal{V}'(H \cup \{A\}))$. A tree formal system (this time unambiguous by definition) is a tuple $\langle \mathsf{TC}', \mathsf{VR}', \mathsf{SA}', \mathrm{Type}', \mathrm{Syn}', \Gamma'\rangle$ where Γ' is a set of statements. The closure of a set $H \subseteq \mathsf{EX}'$ relative to D is defined as in the string case, but the base case instead takes $H \subseteq C$ and $\langle \mathrm{Type}'(T), T\rangle \in C$ for every $T \in \mathsf{ST}'$.

To map an unambiguous formal system $\langle \mathsf{CN}, \mathsf{VR}, \mathrm{Type}, \Gamma, \mathrm{Syn}\rangle$ to a tree formal system $\langle \mathsf{TC}', \mathsf{VR}', \mathsf{SA}', \mathrm{Type}', \mathrm{Syn}', \Gamma'\rangle$, one takes $\mathsf{TC}' = \mathsf{TC}$, VR' to be the set of VH_v singleton trees for $v \in \mathsf{VR}$, $\mathsf{SA}' = \mathsf{SA}$, $\mathrm{Type}'(T) = \mathrm{Type}(T)$, $\mathrm{Syn}' = \mathrm{Syn}$, and $\Gamma' = \{\langle D, \{t(h) \mid h \in H\}, t(A)\rangle \mid \langle D, H, A\rangle \in \Gamma \setminus \mathsf{ST}\}$, where $t(e) = \{\mathrm{Type}(e), \mathrm{Expr}^{-1}(\mathrm{Syn}(e))\}$.

These two formal systems are isomorphic, in the sense that expressions and statements can be mapped freely, respecting the definitions of theorems and axioms, variables and typecodes.

2.5 Models of tree formal systems

Given the isomorphism of the previous section, the model theory of unambiguous formal systems can be mapped to models of tree formal systems, with $\langle \mathsf{U}, \#, \eta\rangle$ satisfying an exactly equivalent set of properties. But the major advantage of the tree formulation is that the substitution property implies that η is completely determined except at syntax axioms, so for trees we will replace η with a new function π.

Definition 9. *Given a function U on TC' satisfying $\mathsf{U}_c \subseteq \mathsf{U}_{\mathrm{Syn}(c)}$, and π a SA-indexed family of functions, where $\pi_a : \prod_i \mathsf{U}_{\mathrm{Type}(v_i^a)} \to \mathsf{U}_{\mathrm{Type}(a)}$ for each $a \in \mathsf{SA}$, define η_μ for $\mu \in \mathsf{VL}$ recursively such that:*

- *For all $v \in \mathsf{VR}$, $\eta_\mu(v) = \mu(v)$.*
- *For each $T \in \mathsf{ST}$ with $a \in \mathsf{SA}$ at the root, $\eta_\mu(T) = \pi_a(\{\eta_\mu(T_i)\}_i)$*
- *For $e = \langle c, T\rangle \in \mathsf{EX}'$, $\eta_\mu(e) = \eta_\mu(T)$ if $\eta_\mu(T) \in \mathsf{U}_c$, otherwise undefined.*

Definition 10. *A model of a tree formal system is a tuple $\langle \mathsf{U}, \#, \pi\rangle$ where U is a function on TC' satisfying $\mathsf{U}_c \subseteq \mathsf{U}_{\mathrm{Syn}(c)}$, and $\#$ is a freshness relation (which is extended from $\bigsqcup_{v\in\mathsf{VT}} \mathsf{U}_c$ to $\bigsqcup_{v\in\mathsf{TC}} \mathsf{U}_c$ by setting $v \# w_c$ iff $v \# w_{\mathrm{Syn}(c)}$ for the copies of w in the disjoint union), and π is a SA-indexed family of functions, where $\pi_a : \prod_i \mathsf{U}_{\mathrm{Type}(v_i^a)} \to \mathsf{U}_{\mathrm{Type}(a)}$ for each $a \in \mathsf{SA}$, such that for each $\mu \in \mathsf{VL}$, and defining η as above:*

- *For each $\langle D, H, A\rangle \in \Gamma'$, if*
  - *$\mu(\alpha) \# \mu(\beta)$ for all $\{\alpha, \beta\} \in D$, and*
  - *$\eta_\mu(h)$ is defined for all $h \in H$,*

  then $\eta_\mu(A)$ is defined.
- *For all $v \in \bigsqcup \mathsf{U}$, $a \in \mathsf{SA}$, and $f \in \prod_i \mathsf{U}_{\mathrm{Type}(v_i^a)}$, if $v \# f_i$ for all i, then $v \# \pi_a(f)$.*

Theorem 2. *Let $\langle \mathsf{U}, \#, \pi\rangle$ be a model for the tree formal system. Then the associated $\langle \mathsf{U}, \#, \eta\rangle$ is a model in the sense of Definition 6.*

Proof.

- The variable application law is true by definition of η, and the axiom application law is true by assumption.

– For the substitution law, suppose σ and $e \in \mathsf{EX}'$ are given. We want to show that $\eta_\mu(\sigma(e)) = \eta_{\sigma(\mu)}(e)$, where $\sigma(\mu) \in \mathsf{VL}$ is defined by $\sigma(\mu)(v) = \eta_\mu(\sigma(v))$. We show this for $\mathrm{Syn}(e) = \langle \mathrm{Type}(T), T\rangle$ by induction on T. In the base case, $T = v$ for $v \in \mathsf{VT}$, so $\eta_{\sigma(\mu)}(v) = \sigma(\mu)(v) = \eta_\mu(\sigma(v))$. Otherwise let $a \in \mathsf{SA}$ be the root of T, so

$$\eta_{\sigma(\mu)}(T) = \pi_a(\{\eta_{\sigma(\mu)}(T_i)\}_i) = \pi_a(\{\eta_\mu(\sigma(T_i))\}_i) = \eta_\mu(\sigma(T)).$$

  Then for $e = \langle c, T\rangle$, if $\eta_{\sigma(\mu)}(T) = \eta_\mu(\sigma(T))$ is in U_c, then both are defined and equal, otherwise both are undefined.
– Dependence on present variables is also provable by induction; in the base case $\eta_\mu(v) = \mu(v)$ only depends on $\mathcal{V}(v) = \{v\}$; and for a tree with $a \in \mathsf{SA}$ at the root, $\eta_\mu(T) = \pi_a(\{\eta_\mu(T_i)\}_i)$ only depends on $\mathcal{V}(T) = \bigcup_i \mathcal{V}(T_i)$. For $e = \langle c, T\rangle$ this property is maintained since $\mathcal{V}(e) = \mathcal{V}(T)$ and $\eta_\mu(e) = \eta_\mu(T)$ or undefined.
– For the freshness substitution law, in the base case $v \# \mu(v) = \eta_\mu(v)$; and for a tree with $a \in \mathsf{SA}$ at the root, if $v \# \mathcal{V}(T)$ then $v \# \mathcal{V}(T_i)$ so $v \# \eta_\mu(T_i)$ for each i, and then by definition $v \# \pi_a(\{\eta_\mu(T_i)\}_i) = \eta_\mu(T)$. For $e = \langle c, T\rangle$ this property is maintained since $\mathcal{V}(e) = \mathcal{V}(T)$ and $\eta_\mu(e) = \eta_\mu(T)$ or undefined.

□

By the isomorphism this approach also transfers to models of unambiguous formal systems. So we are left with the conclusion that a model can be specified by its functions π_a, for each $a \in \mathsf{SA}$; this is known in conventional model theory as the interpretation function.

3 Examples of models

3.1 Propositional logic

We start with a model for classical propositional logic. We define:

$$\mathsf{CN} = \{(,), \to, \neg, \mathrm{wff}, \vdash\}$$

$$\mathsf{VR} = \{\varphi, \psi, \chi, \dots\}$$

$$\Gamma = \{\texttt{wn}, \texttt{wi}, \texttt{ax-1}, \texttt{ax-2}, \texttt{ax-3}, \texttt{ax-mp}\},$$

where the axioms are

– `wn`: wff $\neg\varphi$
– `wi`: wff $(\varphi \to \psi)$
– `ax-1`: $\vdash (\varphi \to (\psi \to \varphi))$
– `ax-2`: $\vdash ((\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi)))$
– `ax-3`: $\vdash ((\neg\varphi \to \neg\psi) \to (\psi \to \varphi))$
– `ax-mp`: $\{\vdash \varphi, \vdash (\varphi \to \psi)\}$ implies $\vdash \psi$

Additionally, $\mathsf{TC} = \{\text{wff}, \vdash\}$ and $\mathsf{VT} = \{\text{wff}\}$ are implied by the preceding definition.

Although the axiom strings appear structured with infix notation, this is not required; we could just as easily have an axiom with string $\vdash)\varphi(\to$. That this is not the case is what makes this a grammatical formal system, with $\text{Syn}(\text{wff}) = \text{Syn}(\vdash) = \text{wff}$. The syntax axioms are $\mathsf{SA} = \{\mathtt{wn}, \mathtt{wi}\}$. In fact this formal system is unambiguous, but we will not prove this here.

This formal system has a nontrivial model:

$$U_{\text{wff}} = \mathsf{Bool} := \{\mathtt{T}, \mathtt{F}\} \quad U_{\vdash} = \{\mathtt{T}\}$$

$$x \# y \text{ is always true}$$

$$\pi_\neg(\mathtt{F}) = \mathtt{T}, \quad \pi_\neg(\mathtt{T}) = \mathtt{F}$$

$$\pi_\to(\mathtt{F},\mathtt{F}) = \mathtt{T}, \quad \pi_\to(\mathtt{F},\mathtt{T}) = \mathtt{T}, \quad \pi_\to(\mathtt{T},\mathtt{F}) = \mathtt{F}, \quad \pi_\to(\mathtt{T},\mathtt{T}) = \mathtt{T}$$

We will write "$\alpha = \mathtt{T}$" simply as "α is true" or "α", treating the elements of Bool as actual truth values in the metalogic. The π_a functions generate η as previously described, so that, for example, if $\mu(\varphi) = \mathtt{T}$, $\mu(\psi) = \mathtt{F}$, then $\eta_\mu(\text{wff } (\neg\varphi \to (\varphi \to \psi))) = \pi_\to(\pi_\neg(\mathtt{T}), \pi_\to(\mathtt{T},\mathtt{F})) = \pi_\to(\mathtt{F},\mathtt{F}) = \mathtt{T}$.

In order to verify that this indeed yields a model, we must check the axiom application law for each non-syntax axiom (usually called "logical axioms" in this context). By our definition of η given π, $\eta_\mu(\langle c, T\rangle)$ is defined iff $\eta_\mu(\langle \text{Syn}(c), T\rangle) \in U_{\text{Syn}(c)}$ (where $\text{Syn}(c) = \text{Type}(T)$); in this case this translates to $\eta_\mu(T) = \mathtt{T}$ since $\text{Syn}(c) = \{\vdash\}$.

- `ax-1`: $\eta_\mu(\vdash (\varphi \to (\psi \to \varphi))) = \pi_\to(\mu_\varphi, \pi_\to(\mu_\psi, \mu_\varphi)) = \mathtt{T}$
- `ax-2`: $\pi_\to(\pi_\to(\mu_\varphi, \pi_\to(\mu_\psi, \mu_\chi)), \pi_\to(\pi_\to(\mu_\varphi, \mu_\psi), \pi_\to(\mu_\varphi, \mu_\chi))) = \mathtt{T}$
- `ax-3`: $\pi_\to(\pi_\to(\pi_\neg(\mu_\varphi), \pi_\neg(\mu_\psi)), \pi_\to(\mu_\psi, \mu_\varphi)) = \mathtt{T}$
- `ax-mp`: If μ_φ and $\pi_\to(\mu_\varphi, \mu_\psi)$ are true, then $\mu_\psi = \mathtt{T}$.

In each case, there are a finite number of variables that range over $\{\mathtt{T}, \mathtt{F}\}$ (such as $\mu_\varphi, \mu_\psi, \mu_\chi$ in the case of `ax-3`), so it suffices to verify that they are true under all combinations of assignments to the variables, i.e. truth table verification.

3.2 MIU system

Let us solve the MU puzzle by using a model. Hofstadter's MIU system [3] is defined in Appendix D of the Metamath book [1], so we will just define the model itself. We have $\mathsf{TC} = \{\text{wff}, \vdash\}$ and $\mathsf{VT} = \{\text{wff}\}$, and let $\text{Syn}(\vdash) = \text{wff}$ with x, y variables of type wff. The axioms are:

- `we`: wff
- `wM`: wff $x\mathtt{M}$
- `wI`: wff $x\mathtt{I}$
- `wU`: wff $x\mathtt{U}$
- `ax`: $\vdash \mathtt{MI}$

- `I_`: $\vdash x\mathtt{I}$ implies $\vdash x\mathtt{IU}$
- `II`: $\vdash \mathtt{M}x$ implies $\vdash \mathtt{M}x\mathtt{M}x$
- `III`: $\vdash x\mathtt{III}y$ implies $\vdash x\mathtt{U}y$
- `IV`: $\vdash x\mathtt{UU}y$ implies $\vdash xy$

The syntax axioms are $\{\mathtt{we}, \mathtt{wM}, \mathtt{wI}, \mathtt{wU}\}$. Note that in `we`, there are no symbols after the typecode, so this says that the empty string is a valid wff. This formal system is weakly grammatical, but not grammatical, because wffs are built from the right, while axiom `II` contains the string $\vdash \mathtt{M}x$, which cannot be parsed as a wff. If there was a syntax axiom "wff xy", then it would be grammatical, but not unambiguous.

In order to solve the MU puzzle, we build the relevant invariant, which is the number of `I`'s modulo 3, into the model. Let $U_{\text{wff}} = \{0, 1, 2\}$, $U_{\vdash} = \{1, 2\}$, let $x \# y$ be always true, and define $\eta_\mu(\text{wff } e)$ to be $\sum_i f(e_i) \bmod 3$, where $f(v) = \mu(v)$ if v is a variable, $f(\mathtt{I}) = 1$, and $f(c) = 0$ for other constants. Let $\eta_\mu(\vdash e) = \eta_\mu(\text{wff } e)$ when $\eta_\mu(\text{wff } e) \in \{1, 2\}$.

Verifying that this yields a model is then equivalent to verifying that the axioms preserve the invariant, and we can deduce that `MU` is not provable because $\eta_\mu(\vdash \mathtt{MU})$ is not defined for any μ.

3.3 Set theory

Of course, the more interesting case is the verification that the full structure of `set.mm` has a nontrivial model. As the background, we need a model of ZFC set theory; let this be $\langle M, \varepsilon\rangle$.

The typecodes are $\mathsf{TC} = \{\text{set}, \text{class}, \text{wff}, \vdash\}$, with the only non-variable typecode being $\vdash$ and $\text{Syn}(\vdash) = \text{wff}$. With these definitions `set.mm` becomes an unambiguous formal system. (Again, the proof of unambiguity is complex and not undertaken here.) Let $V =: U_{\text{set}}$ be any infinite set. This is the set of variables of the "object language" over which Metamath is understood to sit; they are customarily labeled $V = \{v_0, v_1, v_2, \dots\}$. Note that these are *not* actually variables in our sense, they are constants, elements of the set V. Set variables such as x in Metamath range over elements of V.

We will need a few preliminary sets before properly defining U_{wff} and U_{class}. Take $U'_{\text{wff}} = (V \to M) \to \mathsf{Bool}$, that is, the set of functions from $V \to M$ to Bool. The subset of U_{wff} corresponding to true formulas, $U_{\vdash}$, is the singleton $\{\lambda f\, \mathtt{T}\}$ (i.e. the constant function true). Similarly, $U'_{\text{class}} = (V \to M) \to \mathscr{P}(M)$.

Definition 11. *Given $A : (V \to M) \to B$ (A is a wff or class variable) and $W \subseteq V$, we say A* is constant outside W *if for all $f, g : V \to M$, if $f(v) = g(v)$ for all $v \in W$, then $A(f) = A(g)$.*

We define a relation $\#$ on the disjoint union $V \sqcup U'_{\text{wff}} \sqcup U'_{\text{class}}$:

- If $x, y \in V$, then define $x \# y$ iff $x \neq y$.
- If $x \in V$ and $A \in U'_{\text{class}} \sqcup U'_{\text{wff}}$, then define $x \# A$ if A is constant outside $V \setminus \{x\}$.

– The case $A \# x$ when $x \in V$ and $A \in \mathsf{U}_{\text{class}} \sqcup \mathsf{U}_{\text{wff}}$ is covered by symmetry; $x \# y$ is true for any other combination.

To define the real set U_{wff}, we take the set of $A \in \mathsf{U}'_{\text{wff}}$ such that A is "effectively finite-dimensional", that is, A is constant outside some finite $V' \subseteq V$. Similarly, $\mathsf{U}_{\text{class}}$ is the set of effectively finite-dimensional $A \in \mathsf{U}'_{\text{class}}$. There is a minimal set V' outside which A is constant; this set is called $\text{Free}(A)$. It immediately follows from the definition that $A \# x$ for $x \notin \text{Free}(A)$. We can extend the definition slightly to set variables by taking $\text{Free}(x) = \{x\}$ when $x \in V$.

Theorem 3. *$\#$ as defined above is a freshness relation.*

Proof. Clearly $\#$ is a symmetric relation, so we need only verify that for every $c \in \{\text{set}, \text{wff}, \text{class}\}$ and every finite set $W \subseteq \mathsf{VR}$, there is a $v \in \mathsf{U}_c$ with $v \# W$. If $c = \text{wff}$ then take $v = \lambda f\, \mathtt{T}$, and if $c = \text{class}$ then take $v = \lambda f\, M$; in each case $v \# w$ for any $w \in \bigsqcup \mathsf{U}$, so the condition is satisfied.

If $c = \text{set}$, then for each $w \in \mathsf{U}_c$ the set $\text{Free}(w)$ is finite, as is the union $\bigcup_{w \in W} \text{Free}(w)$. Since V is infinite by assumption, choose some $v \in V \setminus \bigcup_{w \in W} \text{Free}(w)$; then for $w \in W$, $v \notin \text{Free}(w)$ implies $v \# w$. □

Definition 12. *For $f : V \to M$, $x \in V$ and $m \in M$, the function $f[x \to m] : V \to M$ is defined by $f[x \to m](y) = f(y)$ for $y \neq x$ and $f[x \to m](x) = m$.*

Finally, we define the π_a functions. By definitional elimination, we can ignore definitional syntax axioms without loss of generality.

– Take $\pi_\neg(f) = f \circ \pi'_\neg$, where $\pi'_\neg$ is the function called $\pi_\neg$ in § 3.1, and similarly for $\pi_\to(f) = f \circ \pi'_\to$.
– $\pi_\forall(x, \varphi)$ is the wff corresponding to $\forall x, \varphi$ and is defined so that $\pi_\forall(x, \varphi)(f)$ iff $\varphi(f[x \to m])$ for all $m \in M$. (Since it comes up often, the restricted quantifier $\forall m \in M$ will be abbreviated $\forall_M m$.)
– The class abstraction, `cab`: class $\{x \mid \varphi\}$, is defined so that $\pi_{\mathtt{cab}}(x, \varphi)(f) = \{m \in M \mid \varphi(f[x \to m])\}$.
– The set-to-class type conversion is a syntax axiom called `cv`: class x. The function for this is defined so that $\pi_{\mathtt{cv}}(x)(f) = \{m \in M \mid m \,\varepsilon\, f(x)\}$.
– Equality of classes is defined by `wceq`: wff $A = B$, and is defined so that $\pi_=(A, B)(f)$ is true iff $A(f) = B(f)$.
– We define $\pi_\in(A, B)(f)$ true if $\exists_M m (A(f) = \{n \in M \mid n \,\varepsilon\, m\} \wedge m \in B(f))$.

For the common case where one or both of the arguments to $=, \in$ are sets, we note that $\pi_\in(\pi_{\mathtt{cv}}(x), A)(f)$ iff $f(x) \in A(f)$, $\pi_\in(\pi_{\mathtt{cv}}(x), \pi_{\mathtt{cv}}(y))(f)$ iff $f(x) \,\varepsilon\, f(y)$, and $\pi_=(\pi_{\mathtt{cv}}(x), \pi_{\mathtt{cv}}(y))(f)$ iff $f(x) = f(y)$. We need $\langle M, \varepsilon \rangle$ to satisfy the extensionality axiom for this to work.

Lemma 1 (The deduction theorem). *$\pi_\to(\varphi, \psi)(f)$ if and only if $\varphi(f)$ implies $\psi(f)$.*

Proof. Suppose not. Then $\pi_\to(\varphi,\psi)(f) = \mathtt{F}$, which by definition implies $\varphi(f) = \mathtt{T}$ and $\psi(f) = \mathtt{F}$, a contradiction. The converse is just `ax-mp` (verified by truth tables). □

Lemma 2 (The non-free predicate). *$x \# \varphi$ iff $\pi_\to(\varphi, \pi_\forall(x,\varphi))(f)$ is true for all f (which is also equivalent to $\eta_\mu(\vdash (\varphi' \to \forall x' \varphi'))$ being defined, where $\mu(\varphi') = \varphi$ and $\mu(x') = x$).*

Proof. By definition, $x \# \varphi$ iff for all $f, g : V \to M$, $f(v) = g(v)$ for all $v \neq x$ implies $\varphi(f) = \varphi(g)$. In this case, given f, and using the deduction theorem, if $\varphi(f)$, then since $f[x \to m]$ differs from f only at x, $\varphi(f[x \to m]) = \varphi(f)$ is true for each m, so $\pi_\forall(x,\varphi)(f)$. Thus $\pi_\to(\varphi, \pi_\forall(x,\varphi))(f)$ by Lemma 1. Conversely, if f, g differ only for $v = x$, either $\varphi(f) = \varphi(g) = \mathtt{F}$, or one of them (say $\varphi(f)$) is true. Then by `ax-mp`, $\pi_\forall(x,\varphi)(f)$, so taking $m = g(x)$, $\varphi(f[x \to g(x)]) = \varphi(g)$ is true. Hence $\varphi(f) = \varphi(g)$, so $x \# \varphi$. □

Theorem 4. *The tuple $\langle U, \#, \eta\rangle$ defined via the above construction is a model for the* `set.mm` *formal system.*

Proof. As in the baby example for propositional calculus, we must verify that η honors all the logical axioms. The tricky ones are the predicate calculus axioms:

- `ax-1,ax-2,ax-3,ax-mp`: Verification by truth tables, as in the propositional calculus example
- `ax-5`: $\vdash (\forall x(\varphi \to \psi) \to (\forall x\varphi \to \forall x\psi))$
  Assume $\pi_\forall(x, \pi_\to(\varphi,\psi))(f)$ and $\pi_\forall(x,\varphi)(f)$; then $\varphi(f[x \to m])$ and $\pi_\to(\varphi,\psi)(f[x \to m])$, so by `ax-mp`, $\psi(f[x \to m])$. Conclude by Lemma 1.
- `ax-6`: $\vdash (\neg\forall x\varphi \to \forall x\neg\forall x\varphi)$
  By Lemma 2, this is equivalent to $x \# \pi_\neg(\pi_\forall(x,\varphi))$. If f, g differ only at x, then $\pi_\forall(x,\varphi)(f)$ if $\varphi(f[x \to m])$ for all m; but $f[x \to m] = g[x \to m]$, so $\pi_\forall(x,\varphi)(f) = \pi_\forall(x,\varphi)(g)$ and $\pi_\neg(\pi_\forall(x,\varphi)(f)) = \pi_\neg(\pi_\forall(x,\varphi)(g))$.
- `ax-7`: $\vdash (\forall x\forall y\varphi \to \forall y\forall x\varphi)$
  By Lemma 1, assume $\pi_\forall(x, \pi_\forall(y,\varphi))(f)$. If $x = y$ then this is the same as $\pi_\forall(y, \pi_\forall(x,\varphi))(f)$, otherwise it reduces to $\varphi(f[x \to m][y \to n])$ for all $m, n \in M$, and note that $f[x \to m][y \to n] = f[y \to n][x \to m]$.
- `ax-gen`: $\vdash \varphi$ implies $\vdash \forall x\varphi$
  If $\varphi(f)$ for all f, then set $f := g[x \to m]$ to deduce $\varphi(g[x \to m])$ for all m, hence $\pi_\forall(x,\varphi)(g)$.
- `ax-8`: $\vdash (x = y \to (x = z \to y = z))$
  By Lemma 1, assume $\pi_=(x,y)$ and $\pi_=(y,z)$. Then $f(x) = f(y) = f(z)$, so $\pi_=(x,z)$.
- `ax-9`: $\vdash \neg\forall x\neg x = y$
  This is equivalent to $\exists m \in M$, $\pi_=(x,y)(f[x \to m])$, or $\exists m \in M$, $m = f[x \to m](y)$. If $x = y$, then any $m \in M$ will do (M is assumed nonempty because it is a model of ZFC), and if $x \neq y$ then take $m = f(y)$.
- `ax-11`: $\vdash (x = y \to (\forall y\varphi \to \forall x(x = y \to \varphi)))$
  Assume $f(x) = f(y)$ and $\forall_M n, \varphi(f[y \to n])$, take some $m \in M$, and assume

$f[x \to m](x) = f[x \to m](y)$. We want to show $\varphi(f[x \to m])$. If $x = y$, then the second hypothesis implies $\varphi(f[y \to m]) = \varphi(f[x \to m])$. Otherwise, $m = f(y) = f(x)$, so $\varphi(f[x \to m]) = \varphi(f) = \varphi(f[y \to m])$.

- `ax-12`: $\vdash (\neg x = y \to (y = z \to \forall x\, y = z))$
  Assume $f(x) \neq f(y) = f(z)$, and take $m \in M$. We want to show $f[x \to m](y) = f[x \to m](z)$. If $x = y$ or $x = z$ then $f(x) = f(y)$ or $f(x) = f(z)$, a contradiction, so $f[x \to m](y) = f(y) = f(z) = f[x \to m](z)$.
- `ax-13`: $\vdash (x = y \to (x \in z \to y \in z))$
  Assume $f(x) = f(y)$ and $f(x)\, \varepsilon\, f(z)$; then $f(y)\, \varepsilon\, f(z)$.
- `ax-14`: $\vdash (x = y \to (z \in x \to z \in y))$
  Assume $f(x) = f(y)$ and $f(z)\, \varepsilon\, f(x)$; then $f(z)\, \varepsilon\, f(y)$.
- `ax-17`: x, φ distinct implies $\vdash (\varphi \to \forall x\, \varphi)$
  This is just the forward direction of Lemma 2.

We can also verify the class axioms:

- `df-clab`: The left hand expression $x \in \{y \mid \varphi\}$ expands to $f(x) \in \{m \in M \mid \varphi(f[y \to m])\}$, that is, $\varphi(f[y \to f(x)])$, while the right side says $f(y) = f(x) \to \varphi(f)$ and $\exists_M m, (f[y \to m](x) = m \wedge \varphi(f[y \to m]))$. If $x = y$, the left conjunct becomes $\varphi(f)$ and the right becomes $\exists_M m, \varphi(f[x \to m])$, which is provable from the other conjunct by setting $m = f(x)$ so that $f[x \to m] = f$. At the same time the left expression also reduces to $\varphi(f[x \to f(x)]) = \varphi(f)$. If $x \neq y$, then $f[y \to m](x) = f(x)$ and the right conjunct becomes $\varphi(f[y \to f(x)])$, and the left conjunct is provable from this since $f(y) = f(x)$ implies $f[y \to f(x)] = f[y \to f(y)] = f$, so both sides are equivalent to $\varphi(f[y \to f(x)])$.
- `df-clel`: We want to show that $\pi_\in(A, B)(f)$ iff there is an m such that $\{n \mid n\, \varepsilon\, m\} = A(f[x \to m])$ and $m \in B(f[x \to m])$, which matches our definition after replacing $A(f[x \to m]) = A(f)$ and $B(f[x \to m]) = B(f)$, since $x \# A$ and $x \# B$.
- `df-cleq`: (This has an extra hypothesis `ax-ext` which is already built into our model.) We want to show that $\pi_=(A, B)(f)$ iff for all m, $m \in A(f[x \to m]) \leftrightarrow m \in B(f[x \to m])$. We are also assuming $x \# A$ and $x \# B$ in this axiom, so this reduces to $m \in A(f) \leftrightarrow m \in B(f)$, or (using extensionality in the metalanguage) $A(f) = B(f)$, which is the definition of $\pi_=(A, B)(f)$.

The "true" axioms of set theory are all phrased in terms of only $=, \in$, and so factor straight through to axioms in $\langle M, \varepsilon \rangle$:

- `ax-ext` (Axiom of Extensionality): The original expression is

$$\pi_\to(\pi_\forall(z, \pi_{\mathtt{wb}}(\pi_\in(z, x), \pi_\in(z, y))), \pi_=(x, y))(f),$$

  which simplifies, according to the definitions, to $\forall_M m(m \varepsilon f(x) \leftrightarrow m \varepsilon f(y)) \to f(x) = f(y)$, for all f. With a change of variables this is equivalent to

$$\forall_M x \forall_M y \forall_M z(z\, \varepsilon\, x \leftrightarrow z\, \varepsilon\, y) \to x = y,$$

exactly the same as the original universally quantified Metamath formula, but with ε in place of $\in$ and $\forall_M$ instead of $\forall$. Since the other axiom expressions are long and the process is similar, I will only quote the final equivalent form after reduction.

- `ax-pow` (Axiom of Power sets): Equivalent to:

$$\forall_M x \exists_M y \forall_M z (\forall_M w (w \,\varepsilon\, z \to w \,\varepsilon\, x) \to z \,\varepsilon\, y)$$

- `ax-un` (Axiom of Union): Equivalent to:

$$\forall_M x \exists_M y \forall_M z (\exists_M w (z \,\varepsilon\, w \wedge w \,\varepsilon\, x) \to z \,\varepsilon\, y)$$

- `ax-reg` (Axiom of Regularity): Equivalent to:

$$\forall_M x (\exists_M y, y \,\varepsilon\, x \to \exists_M y (y \,\varepsilon\, x \wedge \forall_M z (z \,\varepsilon\, y \to \neg z \,\varepsilon\, x)))$$

- `ax-inf` (Axiom of Infinity): Equivalent to:

$$\forall_M x \exists_M y (x \,\varepsilon\, y \wedge \forall_M z (z \,\varepsilon\, y \to \exists_M w (z \,\varepsilon\, w \wedge w \,\varepsilon\, y)))$$

- `ax-ac` (Axiom of Choice): Equivalent to:

$$\forall_M x \exists_M y \forall_M z \forall_M w ((z \,\varepsilon\, w \wedge w \,\varepsilon\, x) \to \exists_M v \forall_M u (\exists_M t (u \,\varepsilon\, w \wedge w \,\varepsilon\, t \wedge u \,\varepsilon\, t \wedge t \,\varepsilon\, y) \leftrightarrow u = v))$$

- `ax-rep`: This one is more complicated than the others because it contains a wff metavariable. It is equivalent to: for all binary relations $\varphi \subseteq M^2$:

$$\forall_M w \exists_M y \forall_M z (\varphi(w, z) \to z = y) \to \exists_M y \forall_M z (z \,\varepsilon\, y \leftrightarrow \exists_M w (w \,\varepsilon\, x \wedge \varphi(w, z))).$$

To show the Metamath form of the axiom from this one, given $\varphi' \in U_{\text{wff}}$ and $f : V \to M$, define $\varphi(w, z) \leftrightarrow \forall_M t, \varphi'(f[w' \to w][y' \to t][z' \to z])$, and apply the stated form of the axiom.

□

Thus if ZFC has a model, so does `set.mm`.

4 Applications of Metamath models

4.1 Independence proofs

We conclude with a few applications of the "model" concept. The primary application of a model is for showing that statements are not provable, because any provable statement must be true in the model. (The converse is not usually true.) This extends to showing that a system is consistent, because any nontrivial model has unprovable statements (and in a logic containing the principle of explosion $\vdash (\varphi \to (\neg\varphi \to \psi))$, this implies that there is no provable statement whose negation is also provable). But it can also be applied for independence

proofs, by constructing a (necessarily nonstandard) model of all statements except the target statement.

For an easy example, if we change the definition of our model of propositional calculus so that instead $\pi_\neg(\mathtt{T}) = \pi_\neg(\mathtt{F}) = \mathtt{T}$, we would have a new model that still satisfies `ax-1`, `ax-2`, and `ax-mp` (because they do not involve $\neg$), but violates `ax-3`. If we take $\mu_\varphi = \mathtt{F}$ and $\mu_\psi = \mathtt{T}$, we get

$$\begin{aligned}\eta_\mu(\text{wff } ((\neg\varphi \to \neg\psi) \to (\psi \to \phi))) &= \pi_\to(\pi_\to(\pi_\neg(\mu_\varphi), \pi_\neg(\mu_\psi)), \pi_\to(\mu_\psi, \mu_\varphi)) \\ &= \pi_\to(\pi_\to(\mathtt{T}, \mathtt{T}), \pi_\to(\mathtt{T}, \mathtt{F})) \\ &= \pi_\to(\mathtt{T}, \mathtt{F}) = \mathtt{F},\end{aligned}$$

so $\eta_\mu(\vdash ((\neg\varphi \to \neg\psi) \to (\psi \to \phi)))$ is not defined. Thus this shows that `ax-3` is not provable from `ax-1`, `ax-2`, `ax-mp` and the syntax axioms alone (although this should not come as a surprise since none of the other axioms use the $\neg$ symbol).

4.2 Gödel's completeness theorem

One important construction that can be done for any arbitrary (string-based) model is to use a formal system as a model of itself. This model will have the property that the theorems (with no hypotheses) are the only statements that are true in the model, leading to an analogue of Gödel's completeness theorem for statements with no hypotheses and all variables distinct. It is also the "original" model of Metamath, from which the terminology "disjoint variable condition" and the "meta" in Metamath are derived.

Given a formal system $\langle \mathsf{CN}, \mathsf{VR}, \text{Type}, \Gamma \rangle$, choose some set VR', with types for each variable, such that $\{v \in \mathsf{VR}' \mid \text{Type}(v) = c\}$ is infinite for each $c \in \mathsf{VT}$. (It is possible to use $\mathsf{VR}' = \mathsf{VR}$ here, provided that VR satisfies this condition, but it is also helpful to distinguish the two "levels" of variable in the contruction.) Using $\mathsf{CN}' = \mathsf{CN}$, define EX' analogously with the new sets. We will call formulas in EX' the "object level" and those in EX the "meta level".

We also define a substitution $\sigma : \mathsf{EX} \to \mathsf{EX}'$ in the same way as § 2.1.8. Here variables of the meta level are substituted with expressions in the object level.

We can build another formal system at the object level, where $\langle D', H', A' \rangle \in \Gamma'$ if there is some $\langle D, H, A \rangle \in \Gamma$ and a substitution $\sigma : \mathsf{EX} \to \mathsf{EX}'$ such that $\forall v \in \mathsf{VR}, \sigma(v) \in \mathsf{VR}'$ (σ substitutes variables for variables) and $\sigma(v) \neq \sigma(w)$ for each $\{v, w\} \in D$, and $D' = \mathsf{DV}'$ (all variables are distinct), $H' = \sigma(H)$ and $A' = \sigma(A)$. This new formal system differs from the original one only in having a different set of variables.

Theorem 5. *Let $A \in \mathsf{U}_c$ if there is some theorem $\langle D, \emptyset, A \rangle$ in the object level formal system with* $\text{Type}(A) = c$*, define $e \# e'$ when $\mathcal{V}(e) \cap \mathcal{V}(e') = \emptyset$, and let η_μ be the unique substitution $\mathsf{EX} \to \mathsf{EX}'$ satisfying $\eta_\mu(\mathsf{VH}_v) = \mu(v)$, restricted to the e such that $\eta_\mu(e) \in \mathsf{U}_{\text{Type}(e)}$. Then $\langle \mathsf{U}, \#, \eta \rangle$ is a model for the meta level formal system $\langle \mathsf{CN}, \mathsf{VR}, \text{Type}, \Gamma \rangle$.*

Proof.

- The type correctness and variable application laws are true by definition, and substitution and dependence on present variables are a consequence of properties of substitutions.
- The relation # is a freshness relation because the finite set $\bigcup_{e \in W} \mathcal{V}(e)$ misses some variable in each type.
- The freshness substitution rule says that if $\mathcal{V}(w) \cap \mathcal{V}(\mu(w)) = \emptyset$ for all $w \in \mathcal{V}(e)$, then $\mathcal{V}(w) \cap \mathcal{V}(\eta_\mu(e)) = \emptyset$, which follows from $\mathcal{V}(\eta_\mu(e)) \subseteq \bigcup_{w \in \mathcal{V}(e)} \mathcal{V}(\mu(w))$ which is a basic property of variables in a substitution.
- The axiom application law translates directly to the induction step of closure in § 2.1.13 for the object level formal system.

□

Corollary 1 (Gödel's completeness theorem). *A statement* $\langle \mathsf{DV}, \emptyset, A \rangle$ *of a formal system is a theorem iff it is true in every model.*

Proof. The forward direction is trivial by the definition of a model. For the converse, a statement true in the model of Theorem 5, with $\mathsf{VR}' \supseteq \mathsf{VR}$ extended to contain infinitely many variables of each type, is derivable by definition. □

Acknowledgments. The author wishes to thank Norman Megill and the anonymous reviewers for pointing out some minor and major omissions in early drafts of this work.

References

1. Megill, N.: Metamath: A Computer Language for Pure Mathematics. Lulu Publishing, Morrisville, North Carolina (2007), `http://us.metamath.org/downloads/metamath.pdf`
2. Tarski, A.: "A Simplified Formalization of Predicate Logic with Identity," *Archiv für Mathematische Logik und Grundlagenforschung*, 7:61-79 (1965).
3. Hofstadter, D.: *Gödel, Escher, Bach.* Basic Books, Inc., New York (1979).