Remark 1.2.15 (Proof systems for FOL over arbitrary vocabularies). The requirement that the axioms and proof rules of Hilbert proof systems are decidable sets of formulas or formula-tuples is not adequate for reasoning about FOL over undecidable vocabularies. To treat FOL over arbitrary (possibly even uncountable) vocabularies and variable-sets, we switch to a slightly more general definition of a Hilbert proof system. In this general setting, the axioms and proof rule are decidable infinite sets of tuples of meta-formulas where the atoms are formula symbols (rather than concrete formulas). Such meta-formulas are generated from

$$\phi \ ::= \ \mathit{true} \ \mid \ \Psi \ \mid \ \phi_1 \wedge \psi_2 \ \mid \ \neg\phi \ \mid \ \forall x.\phi$$

where Ψ ranges over the formula symbols of some decidable, infinite set and x over the elements of some decidable infinite variable-set. Similarly, the axioms and proof rules refer to abstract symbols for terms (rather than concrete terms) and abstract symbols for predicate and function symbols (rather than predicate and function symbols of a concrete vocabulary). The requirement is now that all abstract symbols, i.e., the formula symbols, abstract symbols for the predicate and function of each arity, and the term symbols as well as the variables, are taken from pairwise disjoint decidable infinite sets. Futhermore, decidability of the axioms and proof rules is required. Indeed, this applies to the axioms presented for propositional logic (A1)-(A8) or (PT), the FO-quantifiers (Q1), (Q2) and (Q3) as well as the presented axioms for the equality symbol and the modus ponens.

In this more general setting, $\mathfrak{D}$-proofs are defined as finite sequences over such meta-formulas with abstract symbols imposing the same conditions as in Definition 1.2.1 on page 19. Given a vocabulary of arbitrary cardinality, one can then instantiate the $\mathfrak{D}$-proofs by uniformly substituting the abstract symbols with concrete ones. The remaining concepts (derivation relation, soundness, completeness, and so on) can then be defined as before.

With this more general notion of Hilbert proof systems, FOL over arbitrary vocabularies and variable-sets has sound and complete proof systems. All properties that we will derive from Gödel's completeness theorem for FOL hold for arbitrary vocabularies; the only exception are following considerations on the semi-decidability (see Lemma 1.2.16 below). These results can only be established if the set of all FOL-formulas over the given vocabulary and variable-set are recursively enumerable. ■

Gödel's completeness theorem has several important consequences. A simple observation is the semi-decidability of the set of FOL-tautologies. Recall that semi-decidability of the set of FOL-tautologies means that there is an algorithm that takes as input a FOL-formula ϕ and terminates with the answer "yes" if ϕ is a tautology. Otherwise, i.e., if ϕ is not valid, then the algorithm either halts with the answer "no" or does not terminate (see Figure 9). The existence of such a semi-decision procedure is equivalent to the existence of a recursive enumeration of all valid formulas, i.e., a non-terminating algorithm that outputs all valid FOL-formulas, but no other formula.

Lemma 1.2.16 (Semi-decidability of FOL-tautologies). *The set of all valid FOL-formulas (over some fixed recursively enumerable vocabulary* Voc *and recursively enumerable variable-set* Var*) is recursively enumerable.*

Proof. Let $\mathfrak{D}$ be any sound and weakly complete deductive Hilbert proof system for FOL. Since the axioms and rules of $\mathfrak{D}$ are decidable, there is an algorithmic way to check whether a given

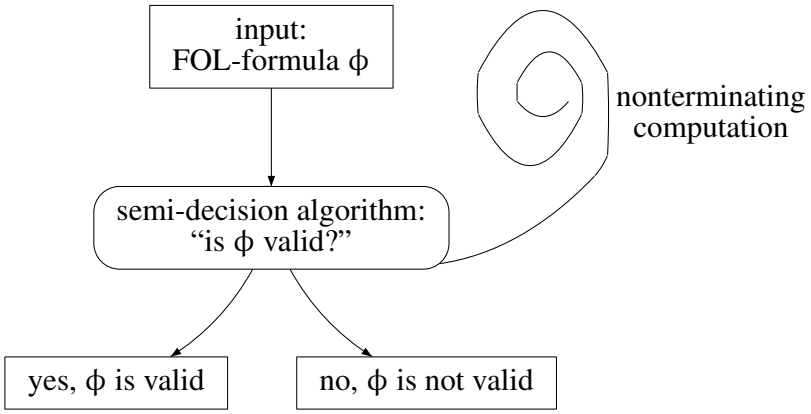

Figure 9: Schema for a semi-decision procedure for FOL tautologies

finite sequence of formulas yields a derivation in $\mathfrak{D}$. As the vocabulary *Voc* and variable-set *Var* are supposed to be recursively enumerable, so is the set of all FOL-formulas over $(\mathit{Voc}, \mathit{Var})$. But then also the set of all finite sequences of FOL-formulas is recursively enumerable.

Hence, an algorithmic enumeration of all valid FOL-formulas is obtained as follows. We generate algorithmically all finite sequences of formulas. For each of these formula-sequences $\psi_1, \ldots, \psi_m$ we apply an algorithm to check whether $\psi_1, \ldots, \psi_m$ is a derivation in $\mathfrak{D}$. If so then the last formula ψ_m will be returned. Otherwise the considered sequence is ignored. In this way, we obtain a (non-terminating) algorithm that outputs exactly the valid formulas. □

Note that the above proof works for any logic that has sound and weakly complete deductive calculi. Thus, the existence of sound and weakly complete deductive calculi implies that the set of all tautologies of that logic is recursively enumerable.

1.3 Undecidability of FOL

The set of all valid FOL-formulas is recursively enumerable, but not decidable. The former is a consequence of the existence of sound and complete deductive calculi for FOL and holds for all vocabularies. This section addresses the undecidability of the validity problem and other decision problems for FOL. Although these undecidability results could be established for any vocabulary with at least one predicate symbol of arity two or more, we will not care about such details here and use several predicate and function symbols. For simplicity, let us suppose the underlying vocabulary contains countably many predicate and function symbols of arbitrary arity. This vocabulary will be denoted by Voc_ω.

Theorem 1.3.1 (Undecidability of FOL). *The following problems for FOL-sentences over* Voc_ω *are undecidable:*

validity problem	"is ϕ valid?"
satisfiability problem	"is ϕ satisfiable?"
equivalence problem	"does $\phi \equiv \psi$ hold?"
consequence problem	"does $\phi \models \psi$ hold?"

Proof. Let us first observe that the undecidability of the validity problem implies undecidability of the other decision problems mentioned above as

$$\phi \text{ is valid iff } \neg\phi \text{ is not satisfiable iff } \mathit{true} \equiv \phi \text{ iff } \mathit{true} \Vdash \phi.$$

Thus, any deterministic algorithm that solves one of these four problems would yield at the same time a procedure to solve the other decision problems. It therefore suffices to establish the undecidability of the validity problem. A comparably simple proof can be provided by means of a reduction from the *Post's correspondence problem* (PCP). (An alternative proof will be given later by providing a reduction from the halting problem for deterministic Turing machines.) The PCP takes as input a finite sequence

$$K = (u_1, v_1), \ldots, (u_n, v_n)$$

of pairs (u_i, v_i) where the u_i's and v_i's are nonempty finite words over $\{0,1\}$. The question is whether there exists a nonempty sequence $i_1, i_2, \ldots, i_k$ of indices $i_1, \ldots, i_k \in \{1, \ldots, n\}$ such that

$$u_{i_1} u_{i_2} \ldots u_{i_k} = v_{i_1} v_{i_2} \ldots v_{i_k}.$$

The PCP is known to be undecidable. The goal is now to provide an algorithmic transformation $K \mapsto \phi_K$ that constructs for a given input K for the PCP a FOL-formula ϕ_K such that the PCP is solvable for K if and only if ϕ_K is valid. For this we deal with a subvocabulary *Voc* of Voc_ω consisting of one binary predicate symbol P, two unary function symbols f_0 and f_1 and a constant symbol c. If $\ell_1, \ldots, \ell_m \in \{0,1\}$ and t is a term then let

$$f_{\ell_1 \ell_2 \ldots \ell_m}(t) \stackrel{\text{def}}{=} f_{\ell_m}(f_{\ell_{m-1}}(\ldots f_{\ell_2}(f_{\ell_1}(t))\ldots)).$$

E.g., $f_{001}(t) = f_1(f_0(f_0(t)))$. The idea is now to encode the words $u_i, v_i \in \{0,1\}^+$ by the terms $f_{u_i}(c)$ and $f_{v_i}(c)$, respectively. The predicate symbol P is used to represent the set of pairs $(u, v) \in \{0,1\}^+ \times \{0,1\}^+$ such that $u = u_{i_1} \ldots u_{i_k}$ and $v = v_{i_1} \ldots, v_{i_k}$ for some nonempty index-sequence $i_1, \ldots, i_k$ with $i_j \in \{1, \ldots, n\}$ for $1 \leqslant j \leqslant k$. Formally, we define the FOL-formula ϕ_K as follows:

$$\phi_K \stackrel{\text{def}}{=} \psi_1 \wedge \psi_2 \rightarrow \theta$$

where ψ_1, ψ_2 and θ are given by:

$$\psi_1 \stackrel{\text{def}}{=} \bigwedge_{1 \leqslant i \leqslant n} P\big(f_{u_i}(c), f_{v_i}(c)\big)$$

$$\psi_2 \stackrel{\text{def}}{=} \forall x \forall y. \big(P(x,y) \rightarrow \bigwedge_{1 \leqslant i \leqslant n} P\big(f_{u_i}(x), f_{v_i}(y)\big)\big)$$

$$\theta \stackrel{\text{def}}{=} \exists x. P(x,x)$$

For example, if $K = (10,0), (01,001)$ then:

$$\psi_1 = P(f_{10}(c), f_0(c)) \wedge P(f_{01}(c), f_{001}(c))$$

$$\psi_2 = \forall x \forall y. (P(x,y) \rightarrow P(f_{10}(x), f_0(y)) \wedge P(f_{01}(x), f_{001}(y)))$$

Of course, given K, formula ϕ_K can be constructed algorithmically. It remains to show that the PCP has a solution for K if and only if ϕ_K is valid.

Let us first suppose that ϕ_K is valid. Let $\mathcal{A}$ be the following structure. The domain of $\mathcal{A}$ is $A = \{0,1\}^*$. The interpretation of the constant symbol c is the empty word, i.e., $c^{\mathcal{A}} = \varepsilon$. The functions $f_0^{\mathcal{A}}, f_1^{\mathcal{A}} : \{0,1\}^* \to \{0,1\}^*$ are given by:

$$f_0^{\mathcal{A}}(w) \stackrel{\text{def}}{=} w0, \qquad f_1^{\mathcal{A}}(w) \stackrel{\text{def}}{=} w1$$

for all $w \in \{0,1\}^*$. The meaning of the predicate symbol P is given by:

$$P^{\mathcal{A}} \stackrel{\text{def}}{=} \big\{ (u_{i_1} \dots u_{i_k}, v_{i_1} \dots v_{i_k}) : i_1, \dots, i_k \in \{1, \dots, n\}, k \geqslant 1 \big\}$$

Let $f^{\mathcal{A}}_{\ell_1 \dots \ell_m} = f^{\mathcal{A}}_{\ell_m} \circ \dots \circ f^{\mathcal{A}}_{\ell_1}$. Then, for all words $w \in \{0,1\}^*$:

$$\begin{aligned} f^{\mathcal{A}}_{\ell_1 \ell_2 \ell_3 \dots \ell_m}(w) &= f^{\mathcal{A}}_{\ell_m}(\dots f^{\mathcal{A}}_{\ell_3}(f^{\mathcal{A}}_{\ell_2}(f^{\mathcal{A}}_{\ell_1}(w)))\dots) \\ &= f^{\mathcal{A}}_{\ell_m}(\dots f^{\mathcal{A}}_{\ell_3}(f^{\mathcal{A}}_{\ell_2}(w\ell_1))\dots) \\ &= f^{\mathcal{A}}_{\ell_m}(\dots f^{\mathcal{A}}_{\ell_3}(w\ell_1\ell_2)\dots) \\ &\vdots \\ &= f^{\mathcal{A}}_{\ell_m}(w\ell_1 \dots \ell_{m-1}) \\ &= w\ell_1 \dots \ell_m \end{aligned}$$

Hence, $f^{\mathcal{A}}_u(w) = wu$ for all words $u, w \in \{0,1\}^*$. As $c^{\mathcal{A}}$ is the empty word, we get:

$$(f_{u_i}(c))^{\mathcal{A}} = u_i \quad \text{and} \quad (f_{v_i}(c))^{\mathcal{A}} = v_i$$

for $1 \leqslant i \leqslant n$. But then:

$$\big((f_{u_i}(c))^{\mathcal{A}}, (f_{v_i}(c))^{\mathcal{A}}\big) = (u_i, v_i) \in P^{\mathcal{A}}$$

Therefore, $\mathcal{A} \models \psi_1$. We also have $\mathcal{A} \models \psi_2$ as

$$(u,v) \in P^{\mathcal{A}} \text{ implies } \big(f^{\mathcal{A}}_{u_i}(u), f^{\mathcal{A}}_{v_i}(v)\big) = (u u_i, v v_i) \in P^{\mathcal{A}}.$$

Thus, $\mathcal{A} \models \psi_1 \wedge \psi_2$. Since ϕ_K is valid this yields that $\mathcal{A} \models \theta$. But then there exists a word $w \in \{0,1\}^*$ with $(w,w) \in P^{\mathcal{A}}$. By definition of $P^{\mathcal{A}}$ this means that there exists a nonempty sequence $i_1, \dots, i_k$ of indices $i_j \in \{1, \dots, n\}$ such that $u_{i_1} \dots u_{i_k} = w = v_{i_1} \dots v_{i_k}$. Hence, $i_1, \dots, i_k$ yields a solution for the PCP for K.

Let us now suppose that the PCP is solvable for K. Let $i_1, \dots, i_k$ be a solution, i.e., $k \geqslant 1$ and

$$u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}.$$

We have to show that ϕ_K is valid. Let $\mathcal{A}$ be a structure with domain A. If $\mathcal{A} \not\models \psi_1 \wedge \psi_2$ then $\mathcal{A} \models \phi_K$. Suppose now that $\mathcal{A} \models \psi_1 \wedge \psi_2$. The goal is to show that $\mathcal{A} \models \theta$. For this, we provide an embedding of $\{0,1\}^*$ in A. Formally, we define a function $e : \{0,1\}^* \to A$ by induction on the length of the words in $\{0,1\}^*$.

- For the empty word ε, we put $e(\varepsilon) \stackrel{\text{def}}{=} c^{\mathcal{A}}$.
- For $w \in \{0,1\}^*$ and $\ell \in \{0,1\}$, we put $e(w\ell) \stackrel{\text{def}}{=} f^{\mathcal{A}}_{\ell}(e(w))$.

We then have $e(w) = (f_w(c))^{\mathcal{A}}$ for all nonempty words w. Since $\mathcal{A} \models \psi_1$ we get:

$$\big(e(u_{i_1}), e(v_{i_1})\big) \;=\; \big((f_{u_{i_1}}(c))^{\mathcal{A}}, (f_{v_{i_1}}(c))^{\mathcal{A}}\big) \in P^{\mathcal{A}}$$

Since $\mathcal{A} \models \psi_2$ we obtain:

$$\big(e(u_{i_1}u_{i_2}), e(v_{i_1}v_{i_2})\big) \;=\; \big((f^{\mathcal{A}}_{u_{i_2}}(e(u_{i_1})), (f^{\mathcal{A}}_{v_{i_2}}(e(v_{i_1}))\big) \in P^{\mathcal{A}}$$

We repeat this argument several times and obtain:

$$\big(e(u_{i_1}u_{i_2}\ldots u_{i_k}), e(v_{i_1}v_{i_2}\ldots v_{i_k})\big) \in P^{\mathcal{A}}$$

Let $a \stackrel{\text{def}}{=} e(u_{i_1}u_{i_2}\ldots u_{i_k})$. As $u_{i_1}\ldots u_{i_k} = v_{i_1}\ldots v_{i_k}$ the above yields $(a,a) \in P^{\mathcal{A}}$. But then $\mathcal{A} \models \exists x.P(x,x) = 0$. □

As a consequence we get that the set of all satisfiable FOL-sentences is not recursively enumerable. If it would be, then a decision algorithm for the validity problem could be obtained as follows. Given a FOL-sentence ϕ, then generate recursively all valid sentences and all satisfiable sentences (in an interleaved way) until either ϕ has been encountered as a valid formula or $\neg\phi$ as a satisfiable formula. In the former case, return "yes", in the latter case return "no". See Figure 10 on page 33. But such a decision procedure for FOL tautologies does not exist.

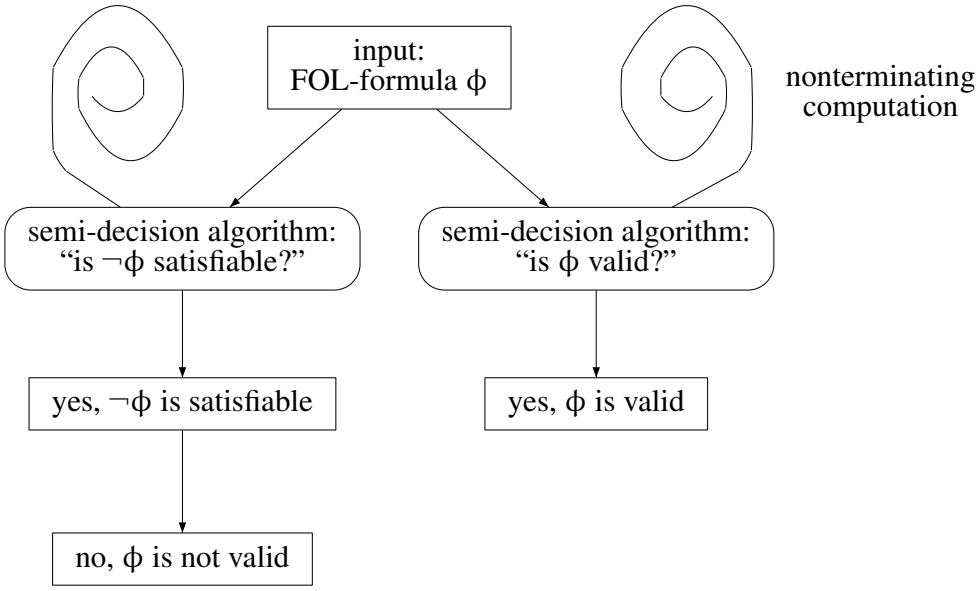

Figure 10: Fictive decision procedure for FOL tautologies

1.3.1 Trakhtenbrot's Theorem

We now will show by a direct reduction from the halting problem for Turing machines that even the *finitary validity problem* "does a given FOL-sentence hold for all finite structures?" is undecidable. Before doing so, let us first observe that the notions validity and satisfiability change when ranging over finite structures only.

Theorem 1.3.2 (Finite model property violated by FOL). *There exist satisfiable FOL-sentences that do not have a finite model, and FOL-sentences that hold for all finite structures, but are not valid.*

Proof. Let P be a binary predicate symbol. We define $\phi \stackrel{\text{def}}{=} \phi_1 \wedge \phi_2 \wedge \neg\phi_3$ where

$$\phi_1 \stackrel{\text{def}}{=} \forall x \exists y.\, \mathsf{P}(x,y)$$

$$\phi_2 \stackrel{\text{def}}{=} \forall x \forall y \forall z.\, (\, \mathsf{P}(x,y) \wedge \mathsf{P}(y,z) \rightarrow \mathsf{P}(x,z)\,)$$

$$\phi_3 \stackrel{\text{def}}{=} \exists x.\, \mathsf{P}(x,x)$$

Sentence ϕ is satisfiable, as the structure $(\mathbb{N},<)$ (i.e., the domain is $\mathbb{N}$ and P is interpreted by the natural order $<$) yields a model for ϕ.

It remains to show that ϕ has no finite model. Let $\mathcal{A}$ be a model for ϕ and $A = \mathit{Dom}^{\mathcal{A}}$. Since $\mathcal{A} \models \phi_1$, there exists a function $f : A \to A$ such that $(a, f(a)) \in \mathsf{P}^{\mathcal{A}}$ for all $a \in A$. (We use here the axiom of choice). Subformula ϕ_2 expresses the transitivity of $\mathsf{P}^{\mathcal{A}}$, while the third subformula $\neg\phi_3 \equiv \forall x.\neg\mathsf{P}(x,x)$ states that the relation $\mathsf{P}^{\mathcal{A}} \subseteq A \times A$ is antireflexive. Let us pick an arbitrary element $a_0 \in A$ (recall that the domain of any structure is nonempty) and define inductively

$$a_{n+1} \stackrel{\text{def}}{=} f(a_n), \quad n = 0,1,2,\ldots$$

then $(a_n, a_m) \in \mathsf{P}^{\mathcal{A}}$ for all $m > n \geqslant 0$ (formula ϕ_2). As $\mathcal{A} \models \neg\phi_3$ we have $a_n \neq a_m$ for $m > n$. Thus, the elements $a_0, a_1, a_2, \ldots$ of A are pairwise distinct. Hence, the domain A of $\mathcal{A}$ is infinite.

An example for a formula that holds for all finite structures, without being valid, is the formula

$$\phi' \stackrel{\text{def}}{=} \phi_1 \wedge \phi_2 \rightarrow \underbrace{\exists x.\mathsf{P}(x,x)}_{=\phi_3}$$

In fact, the above argumentation shows that whenever $\mathcal{A}$ is a finite model for $\phi_1 \wedge \phi_2$ then $\mathcal{A}$ is a model for ϕ_3. Thus, ϕ' holds for all finite structures. On the other hand, $(\mathbb{N},<)$ is an infinite model for $\phi_1 \wedge \phi_2$ where ϕ_3 does not hold. Thus, ϕ' is not valid. □

Thus, the set of all FOL-sentences that are true for all finite structures is a proper superset of the set of all valid FOL-sentences. Similarly, finitary satisfiability (i.e., satisfiability over some finite structure) is weaker than the standard notion of satisfiability.

As for the undecidability of the validity problem, a single binary predicate symbol is sufficient to establish the undecidability of the finitary validity problem *"does ϕ hold for all finite structures?"*. To simplify the proof we will deal again with the vocabulary Voc_ω that contains infinitely many predicate and function symbols of arbitrary arity. Furthermore, we deal with formulas over an infinite set *Var* of variables. As before, Voc_ω and the variable-set *Var* are supposed to be recursively enumerable.

Notation 1.3.3 (FOL-SAT, FOL-VALID, FOL-SAT-FIN and FOL-VALID-FIN). We write FOL-SAT and FOL-VALID to denote the set of all FOL-sentences over Voc_ω and *Var* that are satisfiable or valid, respectively. Let FOL-SAT-FIN denote the set of all FOL-sentences over Voc_ω and *Var* that have at least one finite model. Similarly, FOL-VALID-FIN denotes the set of all FOL-sentences over Voc_ω and *Var* that hold for all finite structures. ■

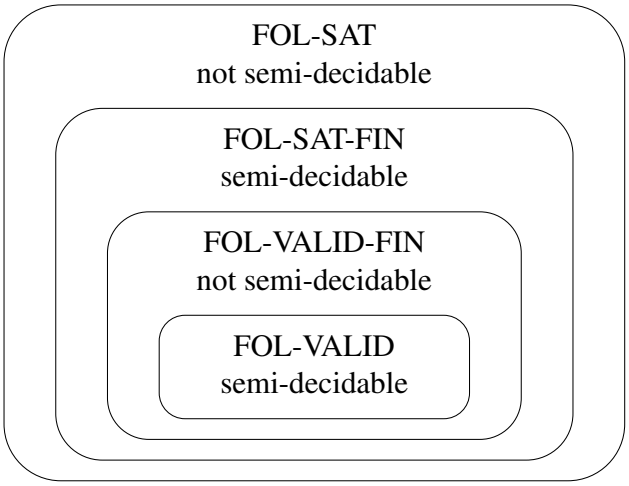

Figure 11: Finitary satisfiability/validity versus standard satisfiability/validity

Figure 11 illustrates the results on the relations between standard satisfiability and validity and finitary satisfiability and validity. We now show that concerning decidability issues, the sets FOL-SAT-FIN and FOL-VALID-FIN behave dually to FOL-SAT and FOL-VALID, respectively. That is, while FOL-SAT is not recursively enumerable, FOL-SAT-FIN has a semi-decision algorithm. Vice versa, while FOL-VALID is recursively enumerable, FOL-VALID-FIN is not.

Lemma 1.3.4. FOL-SAT-FIN *is recursively enumerable.*

Proof. To see that FOL-SAT-FIN is recursively enumerable, regard the set *Fin-Struc* of all structures $\mathcal{A}$ for Voc_ω where $Dom^{\mathcal{A}}$ is $\{0,1,\ldots,n\}$ for some $n \geqslant 0$. Clearly, the set *Fin-Struc* is recursively enumerable. All finite structures are isomorphic to some $\mathcal{A} \in$ *Fin-Struc*. Hence,

$$\phi \in \text{FOL-SAT-FIN} \quad \text{iff} \quad \mathcal{A} \models \phi \text{ for some } \mathcal{A} \in \textit{Fin-Struc}.$$

A semi-decision algorithm for FOL-SAT-FIN is obtained as follows. Let ϕ be a FOL-sentence over Voc_ω. We regard a recursive enumeration $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \ldots$ for *Fin-Struc*. For each $\mathcal{A}_i$, we check whether $\mathcal{A}_i \models \phi$. (This can be done in an algorithmic way since $\mathcal{A}_i$ is finite. We just compute the truth value of ϕ under $\mathcal{A}_i$ by evaluating the subformulas of ϕ recursively, according to the recursive definition of the satisfaction relation $\models$.) If yes then the semi-decision algorithm stops with the answer "yes, ϕ is satisfiable over some finite structure". Since semi-decidability is the same as recursive enumerability, this yields that FOL-SAT-FIN is recursively enumerable. □

Lemma 1.3.5. FOL-SAT-FIN *is undecidable.*

Proof. We provide a reduction from the *halting problem* for deterministic Turing machines (DTM) with the *empty input word*. The latter is known to be undecidable. Let $\mathcal{T}$ be a DTM with a single tape that is unbounded to the left and right. Let Γ be the tape alphabet of $\mathcal{T}$ (with

typical elements $\mathfrak{a}$, $\mathfrak{b}$, $\mathfrak{c}$) and Q the state-space of $\mathcal{T}$ (with typical elements q, p) and

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{-1, 0, +1\}$$

the transition function. Function δ specifies the next state, written letter and movement of the cursor for given state and symbol at the cursor's position on the tape. The movement of the cursor is given by the third component $M \in \{-1, 0, +1\}$ in the triple $\delta(q, \mathfrak{a}) = (p, \mathfrak{b}, M)$. Value $M = -1$ indicates that the cursor moves one position to the left, $M = +1$ means that the cursor moves to the right, while $M = 0$ means that the position of the cursor remains unchanged. Let $\sqcup \in \Gamma$ denote the blank symbol. Furthermore, let q_0 be the initial state of $\mathcal{T}$ and Q_F the set of final states, i.e., the states where the computation halts. Since $\mathcal{T}$ stops as soon as a final state $q \in Q_F$ has been reached, the values $\delta(q, \mathfrak{a})$ of the transition function are irrelevant for $q \in Q_F$. *Configurations* of $\mathcal{T}$ are written in the form

$$w_l\, q\, w_r \quad \text{where } w_l, w_r \in \Gamma^+ \text{ and } q \in Q.$$

The notation $w_l\, q\, w_r$ for a configuration of $\mathcal{T}$ means that the current state is q, the content of the tape is the word $w_l w_r$ and the cursor points to the first letter of w_r. Only tape cells that have been reached so far are taken into account, except for the case where no tape cell on the left of the cursor has been visited so far. In this case, the word w_l consists of the blank symbol $\sqcup$. Let $\vdash_{\mathcal{T}}$ be the one-step configuration relation that formalizes the stepwise behavior of $\mathcal{T}$, i.e.:

$$w_l\, q\, w_r \;\vdash_{\mathcal{T}}\; w_l'\, p\, w_r' \text{ iff } \begin{cases} \mathcal{T} \text{ moves within one step from} \\ \text{configuration } w_l\, q\, w_r \text{ to } w_l'\, p\, w_r' \end{cases}$$

Similarly, for $t \in \mathbb{N}$, the relation $\vdash_{\mathcal{T}}^t$ formalizes the effect of $\mathcal{T}$'s computations along t steps:

$$w_l\, q\, w_r \;\vdash_{\mathcal{T}}^t\; w_l'\, p\, w_r'$$

means that $\mathcal{T}$ moves within t steps from configuration $w_l\, q\, w_r$ to $w_l'\, p\, w_r'$. Formally, for configurations C and C' we have $C \vdash_{\mathcal{T}}^t C'$ iff there exists a sequence $C_0, C_1, \ldots, C_t$ of configurations such that $C_0 = C$, $C_t = C'$ and $C_{i-1} \vdash_{\mathcal{T}} C_i$ for $1 \leqslant i \leqslant t$. The goal is now to define a FOL-sentence $\phi_{\mathcal{T}}$ such that:

$$\begin{array}{lll} \phi_{\mathcal{T}} \in \text{FOL-SAT-FIN} & \text{iff} & \mathcal{T} \text{ halts for the empty word} \\ & \text{iff} & \text{there exist some final state } q \in Q_F, \text{ time bound } t \in \mathbb{N} \\ & & \text{and words } w_l, w_r \in \Gamma^* \text{ s.t. } \sqcup\, q_0 \sqcup \;\vdash_{\mathcal{T}}^t\; w_l\, q\, w_r \end{array}$$

The definition of $\phi_{\mathcal{T}}$ relies on an encoding of the configurations of $\mathcal{T}$ for the empty input word by means of a 4-ary predicate P with the following intuitive meaning:

$$P(t, q, w_l, w_r) \text{ holds} \quad \text{iff} \quad \sqcup\, q_0 \sqcup \;\vdash_{\mathcal{T}}^t\; w_l\, q\, w_r$$

The intended meaning of the atomic formula $P(t, q, w_l, w_r)$ is that $\mathcal{T}$'s configuration after exactly t steps is $w_l\, q\, w_r$ when started with the empty word. To formalize the contents of the tape cells, we use unary function symbols $f_{\mathfrak{a}}$ for all symbols $\mathfrak{a} \in \Gamma$. Furthermore, we need a FOL-representation of a counter for the steps that $\mathcal{T}$ performs. Such a counter is encoded by means of a constant symbol 0 and an unary function symbol *succ*, representing the successor relation on natural numbers, and a binary predicate symbol $<$, representing the natural "less than" order on

$\mathbb{N}$. In addition, we use constant symbols q for the states $q \in Q$. The FOL-formula that specifies $\mathcal{T}$'s behavior for the empty input word has the form

$$\phi_{\mathcal{T}} \stackrel{\text{def}}{=} \underbrace{P(0, q_0, 0, 0)}_{\substack{\text{configuration} \\ \text{at time } 0}} \wedge \psi_{\mathcal{T}}$$

where $\psi_{\mathcal{T}}$ describes $\mathcal{T}$'s behavior:

$$\psi_{\mathcal{T}} \stackrel{\text{def}}{=} \psi_{\text{time}} \wedge \bigwedge_{\substack{q \in Q \setminus Q_F \\ a \in \Gamma}} \psi_{q,a} \wedge \psi_{\text{tape}}$$

The purpose of the subformulas ψ_{time}, $\psi_{q,a}$ and ψ_{tape} is as follows. Formula ψ_{time} serves to encode the step-counter and permits to read $succ(x)$ as $x+1$ (the "next" time point of time point x) provided that the configuration of $\mathcal{T}$ at time point x is not halting. Formula $\psi_{q,a}$ describes the transition of $\mathcal{T}$ for the current state q and symbol $a \in \Gamma$. The last formula ψ_{tape} imposes some side constraints which are necessary to ensure that $\psi_{\mathcal{T}}$ encodes the tape content correctly.

Formula ψ_{time} expresses that the binary predicate symbol $<$ and the unary function symbol *succ* encode the relevant time points of $\mathcal{T}$'s computations. It arises by the conjunction of the following FOL-sentences:

$$\begin{aligned}
&\forall x \forall y \forall z.(x < y \wedge y < z \rightarrow x < z) && \text{(transitivity of } <) \\
&\forall x. \neg (x < x) && \text{(antireflexivity of } <) \\
&\forall x \forall y.(x < y \vee y < x \vee x = y) && \text{(linearity for } <) \\
&\forall x.(0 < x \vee x = 0) && \text{(0 is smallest element)} \\
&\forall x.(x < succ(x) \ \vee \ (\neg \exists z.(x < z) \wedge x = succ(x))) \\
&\forall x \forall z.\big(x < z \rightarrow (z = succ(x) \vee succ(x) < z)\big)
\end{aligned}$$

The last two lines declare that $succ(x)$ is the direct successor of x in the sense that $x < succ(x)$ and there is no element z with $x < z < succ(x)$, unless x is the maximal element. The intuitive meaning of x to be maximal is that at time point x the computation of $\mathcal{T}$ terminates by entering a final state. In this case, we require $x = succ(x)$.

The formulas $\psi_{q,a}$ for the non-final states $q \in Q \setminus Q_F$ and the tape symbols $a \in \Gamma$ encode the triples $\delta(q,a) \in Q \times \Gamma \times \{-1, 0, +1\}$. The precise definition depends on the movement of the cursor:

- If $\delta(q,a) = (p,b,0)$ then

  $$\psi_{q,a} \stackrel{\text{def}}{=} \forall x \forall y \forall z.\big(P(x,q,y,f_a(z)) \ \rightarrow \ x < succ(x) \wedge P(succ(x),p,y,f_b(z))\big),$$

  stating that if at time point x the current configuration is $y\,q\,az$ then the next configuration at time point $succ(x) = x+1$ is $y\,p\,bz$.

- If $\delta(q,a) = (p,b,+1)$ then

  $$\psi_{q,a} \stackrel{\text{def}}{=} \forall x \forall y \forall z.\big(P(x,q,y,f_a(z)) \ \rightarrow \ x < succ(x) \wedge P(succ(x),p,f_b(y),z)\big).$$

  This formula states that if the current configuration at time point x is $y\,q\,az$ then the configuration at time point $succ(x) = x+1$ is $y\,b\,p\,z$.

- Similarly, if $\delta(q,a) = (p,b,-1)$ then

$$\psi_{q,a} \stackrel{\text{def}}{=} \forall x \forall y \forall z. \bigwedge_{c \in \Gamma} \big(P(x,q,f_c(y),f_a(z)) \;\rightarrow\; x < \mathit{succ}(x) \wedge P(\mathit{succ}(x),p,y,f_c(f_b(z))) \big)$$

  This formula declares that if at time point x the current configuration is $y\,c\,q\,az$ then the configuration at time point $\mathit{succ}(x) = x+1$ is $y\,p\,cbz$.

Note that in all three cases the subformula $x < \mathit{succ}(x)$ serves to formalize that $\mathcal{T}$'s computation does not halt when the current state at time point x is non-final (i.e., belongs to $Q \setminus Q_F$).

It remains to provide the definition of ψ_{tape}. The purpose of ψ_{tape} is to ensure a correct representation of the tape contents by the functions f_a:

$$\psi_{\text{tape}} \;\stackrel{\text{def}}{=}\; (0 = f_{\sqcup}(0)) \;\wedge\; \forall y \forall z. \bigwedge_{\substack{a,b \in \Gamma \\ a \neq b}} \big(\text{“}f_a(y),\, f_b(z) \text{ non-maximal”} \;\rightarrow\; f_a(y) \neq f_b(z) \big)$$

$$\wedge \quad \forall x. \exists z. \bigvee_{a \in \Gamma} (x = f_a(z))$$

where “$f_a(y),\, f_b(z)$ non-maximal” stands short for $f_a(y) < \mathit{succ}(f_a(y)) \wedge f_b(z) < \mathit{succ}(f_b(z))$.

The purpose of the subformula $0 = f_{\sqcup}(0)$ is to formalize the fact that each tape cell that has not been visited so far contains the blank symbol $\sqcup$. The second part of the formula on the right serves to state that at any proper time point at most one symbol can be in a tape cell. The third part aims to ensure that each tape cell contains some tape symbol.

The essential properties of our construction are as follows:

$$\phi_{\mathcal{T}} \;\Vdash\; \bigwedge_{q \in Q \setminus Q_F} \forall x \forall y \forall z. \Big(P(x,q,y,z) \;\rightarrow\; x < \mathit{succ}(x) \wedge \bigvee_{p \in Q} \exists y' \exists z'. P\big(\mathit{succ}(x),p,y',z'\big) \Big)$$

and

$$\text{if } \sqcup q_0 \sqcup \;\vdash_{\mathcal{T}}^{t}\; a_1 \ldots a_\ell\, p\, b_1 \ldots b_r \text{ then } \quad \phi_{\mathcal{T}} \;\Vdash\; P(\,\mathit{succ}^t(0),\, p,\, f_{a_\ell \ldots a_1}(0),\, f_{b_1 \ldots b_r}(0)\,)$$

where $f_{c_1 \ldots c_m}(0) \stackrel{\text{def}}{=} f_{c_1}(\,f_{c_2}(\ldots(\,f_{c_m}(0)\,)\ldots)\,)$. More precisely:

(i) We use here a standard concept of computability and treat finite words over an alphabet of finite cardinality k as k-adic numbers. For simplicity, we assume $\Gamma = \{0,1,\ldots,k-1\}$ (where $k \geqslant 2$),[2] $0 = \sqcup$ and $Q = \{0,1,\ldots,n-1\}$, the structure $\mathcal{A}_{\mathcal{T}}$ with $\mathit{Dom}^{\mathcal{A}_{\mathcal{T}}} = \mathbb{N}$ and

$$f_a^{\mathcal{A}_{\mathcal{T}}}(m) = mk + a, \quad \mathit{succ}^{\mathcal{A}_{\mathcal{T}}}(m) = m+1, \quad <^{\mathcal{A}_{\mathcal{T}}} = <, \quad 0^{\mathcal{A}_{\mathcal{T}}} = 0, \quad q^{\mathcal{A}_{\mathcal{T}}} = q,$$

$$P^{\mathcal{A}_{\mathcal{T}}} \;=\; \big\{\; (t,q,(a_1 \ldots a_\ell)_k,(b_r \ldots b_1)_k) \;:\; \sqcup q_0 \sqcup \;\vdash_{\mathcal{T}}^{t}\; a_1 \ldots a_\ell\, q\, b_1 \ldots b_r \;\big\}$$

is a model for $\phi_{\mathcal{T}}$. Since $\mathcal{A}_{\mathcal{T}}$ captures the intuitive meaning of the symbols in the vocabulary, we refer to $\mathcal{A}_{\mathcal{T}}$ as the *canonical model* for $\phi_{\mathcal{T}}$. That is, the word $a_1 \ldots a_\ell \in \Gamma^+$ is identified with the k-adic number

[2] The tape alphabet for any Turing machine contains at least two elements: namely the blank symbol and at least one symbol in the input alphabet.

$$(a_1 \dots a_\ell)_k \;=\; \sum_{j=1}^{\ell} a_j \cdot k^{\ell-j}.$$

Relation $P^{\mathcal{A}_\mathcal{T}}$ contains the tuples $(t, q, w_l, w_r) \in \mathbb{N}^4$ such that after exactly t steps, q is the current state and w_l the tape content on the left of the cursor and w_r the tape content on the right of the cursor (including the letter under the cursor), where w_l and w_r are viewed as k-adic numbers. As stated above, the word $w_l = a_1 a_2 \dots a_\ell$ is identified with the k-adic number $(a_1 a_2 \dots a_\ell)_k$. For technical reasons, the reverse order of the digits is used for the encoding of the word w_r. That is, the semantics of w_r as a k-adic number relies on the interpretation of the symbol under the cursor as the least significant digit. I.e., if $w_r = b_1 b_2 \dots b_r$ then under the structure $\mathcal{A}_\mathcal{T}$ the word w_r is interpreted by:

$$b_1 + \dots + b_{r-1} k^{r-2} + b_r k^{r-1} \;=\; (b_r \dots b_2 b_1)_k$$

But then $mk + a = f_a^{\mathcal{A}_\mathcal{T}}(m) = (b_r \dots b_2\, b_1)_k$ implies $b_1 = a$ and:

$$(b_r \dots b_2 b_1)_k \;=\; a + k \cdot (b_2 + \dots + b_{r-1} k^{r-3} + b_r k^{r-2}) \;=\; f_a^{\mathcal{A}_\mathcal{T}}\big((b_r \dots b_2)_k\big)$$

and there is no other symbol $c \in \Gamma \setminus \{a\}$ such that $(b_r \dots b_1)_k \;=\; f_c^{\mathcal{A}_\mathcal{T}}(h)$ for some h.

The fact that $\mathcal{A}_\mathcal{T} \models \psi_{\text{time}}$ and $\mathcal{A}_\mathcal{T} \models \psi_{\text{tape}}$ is obvious. It remains to show that $\mathcal{A}_\mathcal{T}$ is a model for the formulas $\psi_{q,a}$. Let us consider the case $\delta(q,a) = (p, c, -1)$. Then, in fact:

$$\mathcal{A}_\mathcal{T} \;\models\; \psi_{q,a},$$

To check this, we pick a tuple

$$\big(t,\, q,\, \underbrace{(a_1 \dots a_{\ell-1} a_\ell)_k}_{f_{a_\ell}^{\mathcal{A}_\mathcal{T}}((a_1 \dots a_{\ell-1})_k)},\, \underbrace{(b_r \dots b_2 b_1)_k}_{f_{b_1}^{\mathcal{A}_\mathcal{T}}((b_r \dots b_2)_k)}\big) \;\in\; P^{\mathcal{A}_\mathcal{T}}$$

such that $b_1 = a$ and have to show that

$$\big(\mathit{succ}^{\mathcal{A}_\mathcal{T}}(t),\, p,\, (a_1 \dots a_{\ell-1})_k,\, f_{a_\ell}^{\mathcal{A}_\mathcal{T}}(f_{b_1}^{\mathcal{A}_\mathcal{T}}((b_r \dots b_2)_k))\big) \;\in\; P^{\mathcal{A}_\mathcal{T}}$$

As $\big(t, q, (a_1 \dots a_\ell)_k, (b_r \dots b_1)_k\big) \in P^{\mathcal{A}_\mathcal{T}}$ we have:

$$\sqcup\, q_0 \sqcup \quad \vdash_\mathcal{T}^t \quad a_1 \dots a_\ell\; q\; b_1 b_2 \dots b_r \qquad\qquad \text{where } b_1 = a$$

Since $\delta(q,a) = (p, c, -1)$ we get:

$$a_1 \dots a_{\ell-1} a_\ell\; q\; a\, b_2 \dots b_r \;\vdash_\mathcal{T}\; a_1 \dots a_{\ell-1}\; p\; a_\ell c\, b_2 \dots b_r$$

Therefore:

$$\sqcup\, q_0 \sqcup \;\vdash_\mathcal{T}^{t+1}\; a_1 \dots a_{\ell-1}\, p\;\; q\;\; a_\ell c\, b_2 \dots b_r$$

This yields

$$\begin{aligned}&(\mathit{succ}^{\mathcal{A}_\mathcal{T}}(t), p^{\mathcal{A}_\mathcal{T}}, (a_1 \dots a_{\ell-1})_k, f_{a_\ell}^{\mathcal{A}_\mathcal{T}}(f_c^{\mathcal{A}_\mathcal{T}}(b_2 \dots b_r)_k))\\ &= \;(t+1, p, (a_1 \dots a_{\ell-1})_k, (a_\ell c\, b_2 \dots b_r)_k) \;\in\; P^{\mathcal{A}_\mathcal{T}}.\end{aligned}$$

(ii) If $\mathcal{A}$ is a model for $\phi_\mathcal{T}$ and $\sqcup\, q_0 \sqcup \;\vdash_\mathcal{T}^t\; a_1 \dots a_\ell\, q\, b_1 \dots b_r$ then:

- the elements $\overline{0}^{\mathcal{A}}, \overline{1}^{\mathcal{A}}, \overline{2}^{\mathcal{A}}, \ldots, \overline{t}^{\mathcal{A}}$ where $\overline{0}^{\mathcal{A}} = 0^{\mathcal{A}}$ and $\overline{n+1}^{\mathcal{A}} = \mathit{succ}^{\mathcal{A}}(\overline{n}^{\mathcal{A}})$ are pairwise distinct,
- $\mathcal{A} \models P\big(\mathit{succ}^t(0), q, f_{a_\ell \ldots a_1}(0), f_{b_1 \ldots b_r}(0)\big)$

This can be shown by induction on t.

Using (i) and (ii), we get:

- If $\phi_{\mathcal{T}} \in$ FOL-SAT-FIN then $\phi_{\mathcal{T}}$ has a finite model $\mathcal{A}$. Let A be the domain of structure $\mathcal{A}$. By (ii), if $\mathcal{T}$ performs at least t steps when started with the empty input word then A has at least $t+1$ elements, namely $\overline{0}^{\mathcal{A}}, \overline{1}^{\mathcal{A}}, \ldots, \overline{t}^{\mathcal{A}}$. Thus, the number of steps that $\mathcal{T}$ performs must be finite, since otherwise $\mathcal{A}$ could not be finite. Hence, $\mathcal{T}$ halts for the empty input word.
- Vice versa, suppose that $\mathcal{T}$ halts for the empty input word after exactly t steps. Then, the relation $P^{\mathcal{A}_{\mathcal{T}}}$ for the canonical model $\mathcal{A}_{\mathcal{T}}$ of $\phi_{\mathcal{T}}$ as in (i) is finite and of the form

$$P^{\mathcal{A}_{\mathcal{T}}} \;=\; \big\{(0,\ldots),(1,\ldots),\ldots,(t,\ldots)\big\}.$$

  But then we can derive a finite model $\mathcal{A}$ for $\phi_{\mathcal{T}}$ from $\mathcal{A}_{\mathcal{T}}$ as follows. The domain of structure $\mathcal{A}$ is $A \stackrel{\text{def}}{=} \{0,1,\ldots,M\}$ where M is some natural number such that all components in $P^{\mathcal{A}_{\mathcal{T}}}$ are strictly less than M. Furthermore, we require that $n = |Q| < M$. The interpretations for the predicate und function symbols under $\mathcal{A}$ and $\mathcal{A}_{\mathcal{T}}$ are roughly the same, except that we have to adapt the semantics of of the function symbols f_a and *succ* and the preducate symbol $<$. The interpretations of the function symbols *succ* and f_a is redefined by:

$$\mathit{succ}^{\mathcal{A}}(m) \stackrel{\text{def}}{=} \min\big\{m+1, M\big\}, \quad f_a^{\mathcal{A}}(m) \stackrel{\text{def}}{=} \min\big\{mk+a, M\big\}$$

  The semantics of $<$ in $\mathcal{A}$ is defined by the standard "less-than-relation" on $\{0,1,\ldots,M\}$. The meanings of the constant symbols 0 and the predicate symbol P are unchanged, i.e.,

$$0^{\mathcal{A}} \stackrel{\text{def}}{=} 0^{\mathcal{A}_{\mathcal{T}}} = 0 \quad \text{and} \quad P^{\mathcal{A}} \stackrel{\text{def}}{=} P^{\mathcal{A}_{\mathcal{T}}}$$

  Similarly, $q^{\mathcal{A}} \stackrel{\text{def}}{=} q^{\mathcal{A}_{\mathcal{T}}} = q$ for $q \in Q = \{0,1,\ldots,n-1\}$. Note that by the choice of M we have $P^{\mathcal{A}} \subseteq A^4$ and $Q \subseteq A$. Then, $\mathcal{A}$ and $\mathcal{A}_{\mathcal{T}}$ yield the same truth value for $\phi_{\mathcal{T}}$. Hence, $\mathcal{A}$ is a finite model for $\phi_{\mathcal{T}}$.

This completes the proof that $\mathcal{T}$ halts for the empty input word if and only if $\phi_{\mathcal{T}}$ has a finite model. □

Remark 1.3.6 (Undecidability of FOL). The undecidability of the set of all valid FOL-formulas over Voc_ω can be established by means of a slight variant of the above construction. Here, we consider the FOL-formula

$$\phi_{\text{halt}(\mathcal{T})} \;\stackrel{\text{def}}{=}\; \phi_{\mathcal{T}} \rightarrow \underbrace{\bigvee_{q \in Q_F} \exists x \exists y \exists z.\, P(x,q,y,z)}_{\substack{\text{halting configuration at} \\ \text{some time point } x}}$$

Then, $\phi_{\mathrm{halt}(\mathcal{T})}$ is valid if and only if $\mathcal{T}$ halts for the empty input word. Let us see why.

"$\Longleftarrow$": Suppose $\mathcal{T}$ halts for the empty input word, say $\sqcup\, q_0 \sqcup \;\vdash_{\mathcal{T}}^{t}\; a_1 \ldots a_\ell\, q\, b_1 \ldots b_r$ where q is a halting state, i.e., $q \in Q_F$. The task is to show that $\phi_{\mathrm{halt}(\mathcal{T})}$ is a tautology. We pick some structure $\mathcal{A}$. Clearly, if $\mathcal{A} \not\models \phi_{\mathcal{T}}$ then $\mathcal{A}$ is a model for $\phi_{\mathrm{halt}(\mathcal{T})}$. Suppose now that $\mathcal{A} \models \phi_{\mathcal{T}}$. By (ii) in the proof of Lemma 1.3.5, we get:

$$\mathcal{A} \;\models\; P\big(\mathit{succ}^{t}(0), q, f_{a_\ell \ldots a_1}(0), f_{b_1 \ldots b_r}(0)\big)$$

But then $\mathcal{A} \models \bigvee_{q \in Q_F} \exists x \exists y \exists z.\, P(x, q, y, z)$. This yields $\mathcal{A} \models \phi_{\mathrm{halt}(\mathcal{T})}$.

"$\Longrightarrow$": Let us now assume that $\phi_{\mathrm{halt}(\mathcal{T})}$ is valid. We now regard the canonical model $\mathcal{A}_{\mathcal{T}}$ for $\phi_{\mathcal{T}}$ as in (i) in the proof of Lemma 1.3.5. Since $\mathcal{A}_{\mathcal{T}} \models \phi_{\mathrm{halt}(\mathcal{T})}$ (as $\phi_{\mathrm{halt}(\mathcal{T})}$ is valid) and $\mathcal{A}_{\mathcal{T}} \models \phi_{\mathcal{T}}$ (see (i)), we have:

$$\mathcal{A}_{\mathcal{T}} \;\models\; \bigvee_{q \in Q_F} \exists x \exists y \exists z.\, P(x, q, y, z)$$

The definition of $P^{\mathcal{A}_{\mathcal{T}}}$ then yields that $\sqcup\, q_0 \sqcup \;\vdash_{\mathcal{T}}^{t}\; a_1 \ldots a_\ell\, q\, b_1 \ldots b_r$ for some t, some halting state $q \in Q_F$ and some words $a_1 \ldots a_\ell \in \Gamma^*$ and $b_1 \ldots b_r \in \Gamma^*$. Hence, the computation of $\mathcal{T}$ for the empty input word is terminating. ■