A proof of the not-semidecidability of minimum model entailment

Jesús Héctor Domínguez Sánchez

This note presents a proof of the ***not**-semidecidability* of the minimum entailment relation $\not\approx$ of minimum model reasoning described in [1]. It does so by embedding the *non-halting problem for Turing machines* inside minimum model reasoning. An easy corollary of this result is the non-existence of a proof system for minimum model reasoning that is sound, complete and effective with respect to minimum model semantics.

We use the following definitions regarding computability of sets. A set S is:

Decidable If there is a procedure such that for every x when presented with input x, the procedure outputs `YES` whenever $x \in S$ and it outputs `NO` if $x \notin S$.

Semi-decidable If there is a procedure such that for every x when presented with input x, the procedure outputs `YES` whenever $x \in S$ and it outputs `NO` or runs forever if $x \notin S$.

Therefore, if a set is not semi-decidable then it is impossible to enumerate it algorithmically. We will show that the minimum model entailment relation $\not\approx$ is not semi-decidable for recursively axiomatized theories (when looking at the relation as a set), which means that there could be a case of a theory recursively axiomatized whose minimally entailed set of sentences is not algorithmically enumerable (contrary to the case of first-order logic, where this can always be done for a recursively axiomatized theory).

In order to prove the result, the key idea is to embed a not semi-decidable problem into the logic. It is well known that the complement of the halting problem for Turing machines is not semi-decidable.

The bulk of the proof has already been done: It is a well known result that the *halting problem for Turing machines* can be embedded into first-order logic, see for example [2] under chapter "Undecidability of first-order logic". We will extend this result by showing how can we embed the *complement* of the halting problem for Turing machines inside minimum model reasoning. The known result we will use is the following:

Theorem 1. *Let* M *be a Turing machine and* n *an input string to the machine. Then, there is a finite first-order theory $\Phi_{M,n}$ and a first-order sentence H_M such that:*

$$\text{M } \textit{halts on input } \text{n} \iff \Phi_{M,n} \models H_M$$

where $\models$ is the first-order entailment relation, $\Phi_{M,n}$ can be effectively constructed from the machine M *and input* n*, and H_M can be effectively constructed from the machine* M*.*

Intuitively, the theory $\Phi_{M,n}$ encodes the rules of the machine M and the initial configuration of the tape given the input n. The sentence H_M intuitively states "A halting configuration for machine M exists". So that the theory entails every possible configuration in the execution trace of the machine M given the input n and if one of those configurations is a halting configuration, H_M is entailed.

Observe also that the theorem implies the relation $\models$ is not decidable, because the halting problem for Turing machines is not decidable (i.e. if the relation were decidable, given a machine M and input n, build the theory $\Phi_{M,n}$ and the sentence H_M, which can be algorithmically done, and ask if whether or not $\Phi_{M,n} \models H_M$, this will provide an effective procedure for deciding the halting problem, which is a contradiction).

Remark 1. *In the remainder of this document whenever I mention the theory $\Phi_{M,n}$ and sentence H_M (as described in the above theorem) I will be referring to the constructions defined in [2] which I will sketch in here for clarity.*

In [2] they use the class of Turing machines that only have two symbols as their tape alphabet, "blank" (0) *and "mark"* (1)*. The machine receives only one input, representable as a string of marks, example:*

111111. *The machine has its tape positions indexed by integers* $(\ldots, -2, -1, 0, 1, 2, \ldots)$*, so that at the initial configuration the head is pointing at position* 0 *of the tape, the input is written starting at position* 0 *and continuously to the right of position* 0*, and all other positions have blanks on them. The states of the machine are numbered such that* Q_i *represents state* i *for* $i \geq 0$*. And finally, when started computation, the initial configuration of the machine occurs at time* 0*, the configuration following the initial configuration occurs at time* 1*, and so on.*

The first-order language used by the theory $\Phi_{M,n}$ *and sentence* H_M *contains: the constant* $\mathbf{0}$ *which intuitively denotes the natural number* 0*, the binary predicate* $\mathbf{@}(t, x)$ *which intuitively reads "At time* t *the head is scanning position* x *of the tape"; the binary predicate* $\mathbf{X}(t, x)$ *which intuitively reads "At time* t*, position* x *of the tape is marked, i.e. has a* 1 *on it" so that* $\neg\mathbf{X}(t, x)$ *reads "At time* t*, position* x *of the tape is not marked, i.e. has a* 0*"; the binary predicate* $\mathbf{S}(x, y)$ *which describes the successor relation on the integers, i.e. "*x *and* y *are integers and* y *is the successor of* x*"; for each state* i*, the unary predicate* $\mathbf{Q_i}(t)$ *which intuitively reads "At time* t*, the machine is at state numbered* i*"; and, a binary predicate* $<(x, y)$ *describing the strict order relation on the integers.*

The theory $\Phi_{M,n}$ *is constructed as follows:*

- *Sentences describing properties of the successor relation* $\mathbf{S}(x, y)$ *and how it is connected with the relation* $<(x, y)$ *(I will not describe the sentences in here, as it is not critical for the argument on this document, we only need to know that this is a finite set of sentences, see [2] for the actual sentences).*
- *There is a sentence describing the initial configuration of the machine (i.e. the configuration at time* 0*). For example, if the initial state of the machine is* Q_1 *and the input to the machine is* 111*, then the sentence looks like:*

$$\begin{aligned}
&\mathbf{Q_1}(\mathbf{0}) \wedge \mathbf{@}(\mathbf{0}, \mathbf{0}) \wedge \mathbf{X}(\mathbf{0}, \mathbf{0}) \wedge \\
&\exists y(\mathbf{S}(\mathbf{0}, y) \wedge \mathbf{X}(\mathbf{0}, y)) \wedge \\
&\exists y \exists z(\mathbf{S}(\mathbf{0}, y) \wedge \mathbf{S}(y, z) \wedge \mathbf{X}(\mathbf{0}, z)) \wedge \\
&\forall x((x \neq \mathbf{0} \wedge \neg\mathbf{S}(\mathbf{0}, x) \wedge \neg\exists z(\mathbf{S}(\mathbf{0}, z) \wedge \mathbf{S}(z, x))) \rightarrow \neg\mathbf{X}(\mathbf{0}, x))
\end{aligned}$$

  In other words the sentence says: At time 0 *the machine is at state* Q_1*, the head pointing to position* 0*, positions* 0*,* 1 *and* 2 *are marked and any other position different from* 0*,* 1 *and* 2 *are not marked (Note how we need to emulate the numbers* 1 *and* 2 *using the successor relation, since the language only has symbols for* 0*).*

- *For every rule of the machine not leading to a halting state, there is a sentence describing the rule. For example, if the rule says (only if* Q_4 *is not a halting state): "If in state* Q_2 *you read a blank, then go to state* Q_4*, mark the current tape cell and do not move the head", the sentence describing the rule will be:*

$$\begin{aligned}
&\forall t \forall x((\mathbf{Q_2}(t) \wedge \mathbf{@}(t, x) \wedge \neg\mathbf{X}(t, x)) \rightarrow \\
&\quad \exists u(\mathbf{S}(t, u) \wedge \mathbf{Q_4}(u) \wedge \mathbf{X}(u, x) \wedge \mathbf{@}(u, x) \wedge \\
&\qquad \forall y((y \neq x \wedge \neg\mathbf{X}(t, y)) \rightarrow \neg\mathbf{X}(u, y)) \wedge \\
&\qquad \forall y((y \neq x \wedge \mathbf{X}(t, y)) \rightarrow \mathbf{X}(u, y)) \\
&\quad) \\
&)
\end{aligned}$$

  In other words the sentence says: if the configuration at time t *during the computation of the machine is in state* Q_2 *with the head pointing to a blank, then the machine enters configuration of time* u *(which is the successor of time* t*) satisfying the conclusion of the implication which is*

namely the configuration having current state Q_4, the head pointing to the same position, a mark instead of a blank in the current position and every other tape positions remain unchanged (See [2] for a description of the sentences for every possible form of the machine's rules).

Observe that by construction of the theory $\Phi_{M,n}$, the theory will imply every configuration in the computation history of the machine M under input n (see the proof that this is the case in [2]). The proof of this goes intuitively as follows: starting with the sentence describing the initial configuration (at time 0), instantiate every universal sentence describing a rule with time 0 ($t = \mathbf{0}$) and position 0 ($x = \mathbf{0}$). Since M is a deterministic Turing machine, there will be at most one sentence whose condition of the implication will be true, take the conclusion of the implication as the description of the configuration at time u (which is the successor of 0) and instantiate again the universal sentences with this new time u and the corresponding position of the head (given by the conclusion of the previous rule), and so on.

What happens when at some time t and some position x there are no implications having a true condition among the sentences describing the machine rules? This is a halting situation for machine M. This is exactly what the sentence H_M encodes, it is a disjunction of all halting situations of the machine M (there can only be a finite number of halting situations). A halting situation is when the machine does not have a rule for a particular pair: (state, symbol on the tape). So, if there exists a configuration in the computation history of machine M where a halting situation has been reached, the machine will halt. For example, if the machine does not have a rule for being in state Q_6 reading a mark, then the machine will halt at that point, this is encoded in the sentence $\exists t \exists x(\mathbf{Q_6}(t) \wedge \mathbf{@}(t,x) \wedge \mathbf{X}(t,x))$ (if there is ever a configuration at some time t where the machine is in state Q_6 reading a mark, the sentence will be true), so that if we take a disjunction of all those sentences for every halting situation, we effectively will be saying "there is a halting configuration in the computation history of machine M with input n" which is exactly how H_M is built.

All these considerations make clear that the machine M will halt with input n if and only if $\Phi_{M,n} \models H_M$.

Remark 2. *Although not mentioned explicitly in [2], it is easy to come up with a model for theory $\Phi_{M,n}$. I will describe here the construction of this model. The idea is simply to encode in the structure the computation history of machine M with input n and it will follow naturally that the structure satisfies the theory $\Phi_{M,n}$. We will denote by $\overline{M,n}$ the structure built as follows:*

- *The universe is the set of integers $\mathbb{Z}$.*
- $(a,b) \in \mathbf{S}^{\overline{M,n}} \iff b = a + 1$.
- $(a,b) \in <^{\overline{M,n}} \iff a < b$.
- $(t,x) \in \mathbf{X}^{\overline{M,n}} \iff t \in \mathbb{N}$ *and the configuration at time t in the computation history of M with input n has a mark (i.e. a 1) on position x of the tape.*
- $(t,x) \in \mathbf{@}^{\overline{M,n}} \iff t \in \mathbb{N}$ *and the configuration at time t in the computation history of M with input n has the head pointing to position x of the tape.*
- *For every state Q_i of the machine M (for $i \geq 1$):*

  $t \in \mathbf{Q_i}^{\overline{M,n}} \iff t \in \mathbb{N}$ *and the configuration at time t in the computation history of M with input n has the machine to be in state numbered i.*
- $\mathbf{0}^{\overline{M,n}} = 0$.

We will call $\overline{M,n}$ the "standard model for machine M with input n". The model satisfies all sentences describing the successor relation and the order relation in the theory $\Phi_{M,n}$ (just by construction). It also satisfies the sentence describing the initial configuration of the machine, since, by convention, at time 0 the machine has exactly the initial configuration and this configuration is encoded in the structure by construction. The structure also satisfies all of the sentences describing a rule of the machine, since given one of those sentences and given a configuration at time t in the computation history of the machine which satisfies the conditions of the sentence's implication, the rules of the machine will generate the configuration at time $t + 1$ and since the sentence was constructed using the rules of the machine, the

configuration at time $t+1$ will satisfy the conclusion of the sentence's implication. Therefore, $\overline{M,n} \models \Phi_{M,n}$.

Also, another fact about $\overline{M,n}$ we will generalize later on is the following: M halts on input $n \iff \overline{M,n} \models H_M$. The $\Rightarrow$ direction follows from the fact that if machine halts, then there is a configuration at some time t in the computation history of the machine such that the configuration is in a halting situation; since H_M encodes a disjunction of all halting situations of the machine M, then $\overline{M,n} \models H_M$. The $\Leftarrow$ direction follows from the fact if $\overline{M,n} \models H_M$ then there is some time t in which the machine M is in a halting situation, and therefore, machine M halts on input n.

Theorem 1 provides us with even more information if we use the following proposition which says that first-order entailment is embeddable in minimum model entailment:

Proposition 1. *Let $\mathcal{L}$ be a typed first-order signature. For every $\mathcal{L}$-theory $W \subseteq Sen(\mathcal{L})$ and every $\mathcal{L}$-sentence ψ. If the set of possible observations is empty $\mathcal{O} = \emptyset$ then:*

$$W \models \psi \iff \emptyset \mid\!\approx_W \psi$$

The proof is direct if we use the results in the notes [1] and by observing that $\mathcal{T}(\emptyset) = \emptyset \cup \{\neg\rho \mid \rho \in \mathcal{O} \wedge \emptyset \not\models_W \rho\} = \emptyset$ whenever $\mathcal{O} = \emptyset$.

The proposition then tells us that the relation $\mid\!\approx$ is not decidable since $\models$ is not by Theorem 1. But, we won't stop in here, we want to prove a *stronger* result, namely that the relation $\mid\!\approx$ is not semi-decidable for recursively axiomatizable theories (contrary to the relation $\models$ which *is* semi-decidable for recursively axiomatizable theories).

For that matter, we will embed the *complement* of the halting problem for Turing machines in the minimum model entailment relation with help from the Theorem 1, as the following proposition shows:

Proposition 2. *Let* M *be a Turing machine and* n *an input string to the machine. Then, there is a finite first-order theory $\Phi_{M,n}$ and a first-order sentence H_M such that if the set of possible observations is $\mathcal{O} = \{H_M\}$ then:*

$$\text{M } \textit{DOES NOT halt on input } \text{n} \iff \emptyset \mid\!\approx_{\Phi_{M,n}} \neg H_M$$

where $\mid\!\approx$ is the minimum model entailment relation, $\Phi_{M,n}$ can be effectively constructed from the machine M *and input* n*, and H_M can be effectively constructed from the machine* M*.*

Proof. Let M be a Turing machine and n an input to M. Then, by theorem 1, $\Phi_{M,n}$ and H_M exist. Now, set $\mathcal{O} = \{H_M\}$. We will show both directions.

$\Longrightarrow$. Suppose M does not halt on n. Then, by theorem 1, $\Phi_{M,n} \not\models H_M$. Since $\mathcal{O} = \{H_M\}$ then $\mathcal{T}(\emptyset) = \emptyset \cup \{\neg\rho \mid \rho \in \mathcal{O} \wedge \emptyset \not\models_{\Phi_{M,n}} \rho\} = \{\neg H_M\}$. Therefore $\mathcal{T}(\emptyset) \models \neg H_M$ and by monotonicity of first-order logic $\mathcal{T}(\emptyset) \models_{\Phi_{M,n}} \neg H_M$. So, $\emptyset \mid\!\approx_{\Phi_{M,n}} \neg H_M$.

$\Longleftarrow$. Let $\emptyset \mid\!\approx_{\Phi_{M,n}} \neg H_M$, so $\mathcal{T}(\emptyset) \models_{\Phi_{M,n}} \neg H_M$. Suppose for a contradiction that M DOES halt on input n. Then, by theorem 1, $\Phi_{M,n} \models H_M$. Since $\mathcal{O} = \{H_M\}$ then $\mathcal{T}(\emptyset) = \emptyset \cup \{\neg\rho \mid \rho \in \mathcal{O} \wedge \emptyset \not\models_{\Phi_{M,n}} \rho\} = \emptyset$ (since $\Phi_{M,n} \models H_M$). Therefore, $\emptyset \models_{\Phi_{M,n}} \neg H_M$ (since $\mathcal{T}(\emptyset) \models_{\Phi_{M,n}} \neg H_M$). This shows that theory $\Phi_{M,n}$ implies both H_M and $\neg H_M$ and so is inconsistent. But, by remark 2, $\overline{M,n}$ is a model for theory $\Phi_{M,n}$, and so, we have arrived to a contradiction. Therefore, machine M does not halt on input n. □

So the relation $\mid\!\approx$ is not semi-decidable. For suppose it is. Then, given an arbitrary Turing machine M and an arbitrary input n, build the theory $\Phi_{M,n}$ (which can be done effectively) and the sentence H_M (which can be done effectively) as in the above proposition. Set $\mathcal{O} = \{H_M\}$. Then, by the above proposition, query if $\emptyset \mid\!\approx_{\Phi_{M,n}} \neg H_M$ holds. All these steps will give us a semi-decidable procedure for testing that machine M does not halt on input n (since the relation $\mid\!\approx$ is semi-decidable by assumption) and therefore the complement of the halting problem for Turing machines would be semi-decidable, which is a contradiction.

So, we now know that the relation $\mid\!\approx$ is not decidable and not semi-decidable. What about $\not\mid\!\approx$? Could it be semi-decidable? By using the contrapositive of theorem 1 and the contrapositive of proposition 1 we get:

Proposition 3. *Let* M *be a Turing machine and* n *an input string to the machine. Then, there is a finite first-order theory $\Phi_{M,n}$ and a first-order sentence H_M such that if the set of possible observations is $\mathcal{O} = \emptyset$ then:*

$$\text{M } \textit{DOES NOT halt on input } \text{n} \iff \emptyset \not\approx_{\Phi_{M,n}} H_M$$

where $\not\approx$ *is the minimum model entailment relation,* $\Phi_{M,n}$ *can be effectively constructed from the machine* M *and input* n*, and* H_M *can be effectively constructed from the machine* M*.*

Therefore, by a similar argument, the relation $\not\approx$ is not semi-decidable neither.

Proposition 2 has a simple corollary. This result was a conjecture in my masters dissertation (in Chapter 3 of my masters dissertation).

Corollary 1. *There is no proof system* Υ^X *such that for any typed, countable, first-order signature* $\mathcal{L}$*, any* $W \subseteq Sen(\mathcal{L})$*, any* $\mathcal{O} \subseteq Sen(\mathcal{L})$ *and any finite* $\Delta \subseteq \mathcal{O}$*:*

- *If* W *and* $\mathcal{O}$ *are recursive sets, then* Υ^X *is effective (i.e. the set* $Prov_{W,\mathcal{O},\Delta}$ *is recursive where* W*,* $\mathcal{O}$ *and* Δ *act as parameters of proof expressions of the form* $\mathcal{O}, \Delta, \Gamma_1 \mid\!\!\sim^X_W \psi_1 \ \ldots \ \mathcal{O}, \Delta, \Gamma_n \mid\!\!\sim^X_W \psi_n$ *such that for* $1 \leq i \leq n$*,* $\Gamma_i \subseteq Form(\mathcal{L})$ *are finite and* $\psi_i \in Form(\mathcal{L})$*).*
- Υ^X *is sound and complete with respect to minimum model semantics (i.e. for any* $\psi \in Sen(\mathcal{L})$*:* $\mathcal{O}, \Delta, \emptyset \vdash^X_W \psi \iff \Delta \not\approx_W \psi$*)*

Proof. Assume such proof system Υ^X exists. Let M be an arbitrary Turing machine and n an arbitrary input to it. Compute the theory $\Phi_{M,n}$ and the sentence H_M (Both effectively computable) whose existence is implied by proposition 2. By our assumption, set $\mathcal{L}$ to be the language of $\Phi_{M,n} \cup \{H_M\}$ (the language will use only one type). Set $W = \Phi_{M,n}$. Set $\mathcal{O} = \{H_M\}$. Set $\Delta = \emptyset$. Observe that $\Phi_{M,n}$ and $\{H_M\}$ are finite sets, therefore W and $\mathcal{O}$ are recursive sets. So, by assumption, Υ^X is effective. Now, by soundness and completeness assumption of Υ^X we get $\{H_M\}, \emptyset, \emptyset \vdash^X_{\Phi_{M,n}} \neg H_M \iff \emptyset \not\approx_{\Phi_{M,n}} \neg H_M$. Now, by proposition 2 we get:

$$\{H_M\}, \emptyset, \emptyset \vdash^X_{\Phi_{M,n}} \neg H_M \iff M \text{ DOES NOT halt on input } n$$

So, we were able to embed the non-halting problem for Turing machines in the proof system Υ^X. And since the proof system Υ^X is effective, the following will provide a semi-decidable procedure for the non-halting problem for Turing machines (which is a contradiction):

{Let M be a machine and n an input to the machine}
Compute the theory $\Phi_{M,n}$ and the sentence H_M.
$W \leftarrow \Phi_{M,n}$; $\mathcal{O} \leftarrow \{H_M\}$; $\Delta \leftarrow \emptyset$.
for all syntactically well-formed proof expressions Ω in the system Υ^X **do**
  if Ω is a proof for $\neg H_M$ (This condition is decidable since the set $Prov_{W,\mathcal{O},\Delta}$ is recursive as Υ^X is effective by assumption) **then**
    Answer `YES` (i.e. Machine M DOES NOT halt on input n by above biconditional)
  end if
end for

□

In my dissertation, the above result and some other results in Chapter 3 were conditioned under the existence of certain sets. It turns out that a slight generalization of the Turing machine construction of remarks 1 and 2 allows us to prove the existence of those sets. The idea is to generalize the constructed theory such that it will be able to *index every possible Turing machine.*

Instead of using the general halting problem for Turing Machines, we will use the halting problem for Turing Machines with blank input (as this simplifies the matter a lot). The *complement* of the halting problem for Turing machines with blank input is also known to be not semi-decidable (the proof of this goes by embedding the general halting problem into the blank-input halting problem).

Let **M** be a sequence, indexed by the natural numbers, of all possible Turing machines: M_0, M_1, M_2, ... The sequence can be effectively computed and there is an algorithm that given a natural number i returns the machine at position i in the sequence.

We will modify the language of the theory in remark 1 to include the following predicates:

- Constant **0** which intuitively denotes the natural number 0.

- Ternary predicate $\mathbf{@}(i,t,x)$ which intuitively reads "Machine with index i, at time t in its computation history (started with blank input) has its head scanning position x of its tape".
- Ternary predicate $\mathbf{X}(i,t,x)$ which intuitively reads "Machine with index i, at time t in its computation history (started with blank input) has position x of its tape marked, i.e. has a 1 on it" so that $\neg\mathbf{X}(i,t,x)$ reads "Machine with index i, at time t in its computation history (started with blank input) has position x of its tape not marked, i.e. has a 0 on it".
- Binary predicate $\mathbf{S}(x,y)$ which describes the successor relation on the integers, i.e. "x and y are integers and y is the successor of x".
- Binary predicate $<(x,y)$ describing the strict order relation on the integers.
- For each natural number s, the binary predicate $\mathbf{Q_s}(i,t)$ which intuitively reads "Machine with index i, at time t in its computation history (started with blank input) is at state numbered s".

Then, using an almost identical construction as in remark 1, the theory is constructed as follows (we will denote this theory by $\mathcal{T}$):

- As in remark 1, $\mathcal{T}$ has sentences describing properties of the successor relation $\mathbf{S}(x,y)$ and how it is connected with the relation $<(x,y)$. These sentences are identical as those in remark 1.
- For every Turing machine in the sequence $\mathbf{M}$, there is a sentence describing the initial configuration of the machine (i.e. the configuration at time 0). For example, if the initial state of the machine M_0 is Q_1 and the initial state of the machine M_1 is Q_4 (remember that we are using blank inputs for all the machines), then the sentences look like:

$$\mathbf{Q_1}(\mathbf{0},\mathbf{0}) \wedge \mathbf{@}(\mathbf{0},\mathbf{0},\mathbf{0}) \wedge \forall x\, \neg\mathbf{X}(\mathbf{0},\mathbf{0},x)$$
$$\exists i(\mathbf{S}(\mathbf{0},i) \wedge \mathbf{Q_4}(i,\mathbf{0}) \wedge \mathbf{@}(i,\mathbf{0},\mathbf{0}) \wedge \forall x\, \neg\mathbf{X}(i,\mathbf{0},x))$$

  In other words the first sentence says: At time 0 the machine M_0 is at state Q_1, the head pointing to position 0, and its entire tape has only blanks. The second sentence reads: At time 0 the machine M_1 (notice how we needed to emulate number 1 using the successor relation, since the language only has symbols for 0) is at state Q_4, the head pointing to position 0, and its entire tape has only blanks.
- For every Turing machine in the sequence $\mathbf{M}$, and every rule of the machine not leading to a halting state, there is a sentence describing the rule. For example, if a rule for machine M_1 says (only if Q_4 is not a halting state for machine M_1): "If in state Q_2 you read a blank, then go to state Q_4, mark the current tape cell and do not move the head", the sentence describing the rule for machine M_1 will be:

$$\begin{array}{l}
\exists i(\mathbf{S}(\mathbf{0},i)\wedge \\
\forall t\forall x((\mathbf{Q_2}(i,t) \wedge \mathbf{@}(i,t,x) \wedge \neg\mathbf{X}(i,t,x)) \rightarrow \\
\quad \exists u(\mathbf{S}(t,u) \wedge \mathbf{Q_4}(i,u) \wedge \mathbf{X}(i,u,x) \wedge \mathbf{@}(i,u,x)\wedge \\
\qquad \forall y((y \neq x \wedge \neg\mathbf{X}(i,t,y)) \rightarrow \neg\mathbf{X}(i,u,y)) \wedge \\
\qquad \forall y((y \neq x \wedge \mathbf{X}(i,t,y)) \rightarrow \mathbf{X}(i,u,y)) \\
\quad) \\
) \\
)
\end{array}$$

In other words the sentence says: if the configuration at time t during the computation history of machine M_1 has the machine in state Q_2 with the head pointing to a blank, then the machine enters configuration at time u (which is the successor of time t) satisfying the conclusion of the implication which is namely the configuration having current state Q_4, the head pointing to the same position, a mark instead of a blank in the current position and every other tape positions remain unchanged. Observe how we needed to emulate number 1 (for referring to machine M_1) using the successor relation.

Observe that by construction of the theory $\mathcal{T}$, the theory will imply every configuration in the computation history of every machine in the sequence $\mathbf{M}$ (when given to each one of them a blank input). The proof of this goes intuitively as follows: Given an arbitrary Turing machine index i, starting with the sentence describing the initial configuration (at time 0) of machine M_i, instantiate the universal variables on the implication of every sentence describing a rule of machine M_i such that the instantiation refer to the initial configuration of machine M_i, i.e. set $t = \mathbf{0}$ and $x = \mathbf{0}$. Since every machine in $\mathbf{M}$ is a deterministic Turing machine, there will be at most one sentence for machine M_i whose condition of the implication will be true, take the conclusion of the implication as the description of the configuration of machine M_i at time u (which is the successor of 0) and instantiate again the universal sentences with this new time u and the corresponding position of the head (given by the conclusion of the previous rule), and so on.

For building a sentence describing the existence of a halting configuration in the computation history of a machine we extend the construction of the sentence H_M of remark 1 in the natural way, we will denote by H_i the disjunction of all halting situations of machine M_i (relative to index i). For example: If the only halting situations for machine M_1 are the following two: "There is no rule stating what to do when in state Q_0 reading a blank, and there is no rule stating what to do when in state Q_4 reading a mark", H_1 will look like:

$$
\begin{array}{l}
\exists i(\mathbf{S}(\mathbf{0}, i) \wedge (\\
\quad \exists t \exists x(\mathbf{Q_0}(i, t) \wedge \mathbf{@}(i, t, x) \wedge \neg \mathbf{X}(i, t, x)) \vee \\
\quad\quad \exists s \exists y(\mathbf{Q_4}(i, s) \wedge \mathbf{@}(i, s, y) \wedge \mathbf{X}(i, s, y)) \\
))
\end{array}
$$

Following the construction, it is easy to come up with a model for $\mathcal{T}$ by extending the construction of remark 2. Just build a structure encoding the computation history of every machine in the sequence $\mathbf{M}$.

We will denote by $\overline{\mathcal{T}}$ the structure built as follows:

- The universe is the set of integers $\mathbb{Z}$.
- $(a, b) \in \mathbf{S}^{\overline{\mathcal{T}}} \iff b = a + 1$.
- $(a, b) \in <^{\overline{\mathcal{T}}} \iff a < b$.
- $(i, t, x) \in \mathbf{X}^{\overline{\mathcal{T}}} \iff t, i \in \mathbb{N}$ and the configuration at time t in the computation history of machine M_i with blank input has a mark (i.e. a 1) on position x of the tape.
- $(i, t, x) \in \mathbf{@}^{\overline{\mathcal{T}}} \iff t, i \in \mathbb{N}$ and the configuration at time t in the computation history of machine M_i with blank input has the head pointing to position x of the tape.
- For every state predicate Q_s (for $s \geq 1$):

  $(i, t) \in \mathbf{Q_s}^{\overline{\mathcal{T}}} \iff t, i \in \mathbb{N}$ and the configuration at time t in the computation history of machine M_i with blank input has the machine in state numbered s.
- $\mathbf{0}^{\overline{\mathcal{T}}} = 0$.

The model $\overline{\mathcal{T}}$ satisfies all sentences describing the successor relation and the order relation in the theory $\mathcal{T}$ (just by construction). It also satisfies every sentence describing the initial configuration of every machine, since, by convention, at time 0 each machine has exactly the initial configuration and this configuration is encoded in the structure by construction. The structure also satisfies all sentences describing a rule for every machine, since given one of those sentences and given a configuration at time t in the computation history of machine i which satisfies the conditions of the sentence's implication, the rules of machine i will generate the configuration at time $t+1$ and since the sentence was constructed using the rules of the machine, the configuration at time $t+1$ will satisfy the conclusion of the sentence's implication. Therefore, $\overline{\mathcal{T}} \models \mathcal{T}$.

Again, we have the following property of the constructed structure:

Proposition 4. *For every machine M_i in the sequence* **M**.

$$M_i \text{ halts on a blank input} \iff \overline{\mathcal{T}} \models H_i.$$

Proof. $\Longrightarrow$. If machine halts, then there is a configuration at some time t in the computation history of the machine such that the configuration is in a halting situation; since H_i encodes a disjunction of all halting situations of the machine M_i, then $\overline{\mathcal{T}} \models H_i$.

$\Longleftarrow$. If $\overline{\mathcal{T}} \models H_i$ then there is some time t in which the machine M_i is in a halting situation, and therefore, machine M_i halts on a blank input. □

All these arguments allow us to show:

Proposition 5. *For every machine M_i in the sequence* **M**.

$$\mathrm{M_i} \text{ halts on a blank input} \iff \mathcal{T} \models H_i$$

Proof. $\Longrightarrow$. Follows by the argumentation explaining how the theory $\mathcal{T}$ implies the configuration at every time of every machine.

$\Longleftarrow$ (By contrapositive). Suppose M_i does not halt on a blank input. Then, $\overline{\mathcal{T}} \not\models H_i$ (By Proposition 4). Therefore, we have a structure (namely $\overline{\mathcal{T}}$) satisfying $\mathcal{T}$ but not H_i, therefore $\mathcal{T} \not\models H_i$ as desired. □

From now on, we will denote by $\mathcal{H}$ the set $\{H_i \mid i \geq 0\}$, in other words, is the set of all sentences claiming the existence of a halting configuration for every machine in the sequence **M** when started with blank input.

Proposition 6. *$\mathcal{T}$ and $\mathcal{H}$ are recursive sets.*

Proof. The following is a decision procedure for $\mathcal{T}$ and $\mathcal{H}$.

INPUT: a string ϕ.
if ϕ is a well-formed sentence **then**
  $i \leftarrow 0$
  if ϕ has the form: $\exists x_1(\mathbf{S}(\mathbf{0}, x_1) \wedge (\ldots))$ **then**
    $i \leftarrow 1$
  else if ϕ has the form: $\exists x_1, \ldots, x_{n-1}, x_n(\mathbf{S}(\mathbf{0}, x_1) \wedge \ldots \wedge \mathbf{S}(x_{n-1}, x_n) \wedge (\ldots))$ (for $n \geq 2$) **then**
    $i \leftarrow n$
  end if
  Obtain machine with index i from the sequence **M** (This step is recursive)
  if ϕ has the form of the sentence describing the initial configuration of machine i (verifiable by construction of sentences of theory $\mathcal{T}$) **then**
    ANSWER: $\phi \in \mathcal{T}$ and $\phi \notin \mathcal{H}$
  end if
  for all rule r in machine i **do**
    if ϕ implements rule r (verifiable by construction of sentences of theory $\mathcal{T}$) **then**
      ANSWER: $\phi \in \mathcal{T}$ and $\phi \notin \mathcal{H}$
    end if
  end for
  if ϕ has the form of a disjunction of halting situations of machine i (verifiable by construction of sentences in $\mathcal{H}$) **then**

ANSWER: $\phi \notin \mathcal{T}$ and $\phi \in \mathcal{H}$
end if
end if
ANSWER: $\phi \notin \mathcal{T}$ and $\phi \notin \mathcal{H}$ (The case $\phi \in \mathcal{T}$ and $\phi \in \mathcal{H}$ cannot happen because $\mathcal{T}$ and $\mathcal{H}$ are disjoint by construction).

□

Now, we have all the elements to show the following corollary:

Corollary 2. *There is a typed, countable, first-order signature $\mathcal{J}$, recursive set $R \subseteq Sen(\mathcal{J})$, recursive set $\mathcal{P} \subseteq Sen(\mathcal{J})$ and finite $\Lambda \subseteq \mathcal{P}$ such that there is a minimum model in $\mathsf{Mod}_R(\Lambda)$ and $\mathcal{P} \cap \{\psi \in Sen(\mathcal{J}) \mid \Lambda \nvDash_R \psi\}$ is NOT recursively enumerable.*

Proof. Set $\mathcal{J}$ to be the language used by the theory $\mathcal{T}$ (with only one type). Set R to be $\mathcal{T}$. Set $\mathcal{P}$ to be $\mathcal{H}$. Set Λ to be empty $\emptyset$. We will show $\overline{\mathcal{T}}$ is a minimum model in $\mathsf{Mod}_{\mathcal{T}}(\emptyset)$. For $\overline{\mathcal{T}}$ to be a minimum model it needs to be a model of $\mathcal{R} \cup \Lambda \cup \{\neg\phi \mid \phi \in \mathcal{P} \wedge \Lambda \nvDash_{\mathcal{R}} \phi\} = \mathcal{T} \cup \{\neg\phi \mid \phi \in \mathcal{H} \wedge \emptyset \nvDash_{\mathcal{T}} \phi\}$. We know $\overline{\mathcal{T}} \models \mathcal{T}$, so let $\alpha \in \{\neg\phi \mid \phi \in \mathcal{H} \wedge \emptyset \nvDash_{\mathcal{T}} \phi\}$ so $\alpha = \neg\phi$ for some $\phi \in \mathcal{H}$ and $\mathcal{T} \nvDash \phi$. Therefore $\phi = H_i$ for some machine i in the sequence $\mathbf{M}$. By Proposition 5, machine M_i does not halt on a blank input (since $\mathcal{T} \nvDash \phi$). Then, by Proposition 4 $\overline{\mathcal{T}} \nvDash \phi$ which implies $\overline{\mathcal{T}} \models \neg\phi$ and so $\overline{\mathcal{T}} \models \alpha$ as desired.

Now, to show $\mathcal{P} \cap \{\psi \in Sen(\mathcal{J}) \mid \Lambda \nvDash_R \psi\} = \mathcal{H} \cap \{\psi \in Sen(\mathcal{J}) \mid \mathcal{T} \nvDash \psi\}$ is NOT recursively enumerable, assume it is. Then, it is the range of a recursive function f with domain the natural numbers $\mathbb{N}$ (i.e. function f is enumerating the elements of the set using the elements of $\mathbb{N}$ as indices). Then, we can build recursive function g as follows:

$$g(i) = \text{machine in the sequence } \mathbf{M} \text{ the sentence } f(i) \text{ refers to}$$

Remember that by construction we can extract effectively from every sentence in $\mathcal{H}$ the machine the sentence refers to (see for example proof of Proposition 6). Therefore, recursive function g is enumerating the set $\{m \in \mathbf{M} \mid m \text{ DOES NOT halt when started on a blank input}\}$ which is a contradiction, as the complement of the halting problem for Turing machines with blank input is not semi-decidable. To see why g enumerates such set, observe that $f(i)$ is a sentence in $\mathcal{H} \cap \{\psi \in Sen(\mathcal{J}) \mid \mathcal{T} \nvDash \psi\}$ and therefore by Proposition 5, the machine the sentence $f(i)$ refers to, does not halt on a blank input. □

This last corollary provides another proof for Proposition 1 by using the results of Chapter 3 on my masters dissertation. The corollary also provides the required ingredient to prove all results in Chapter 3 of my masters dissertation (Since the corollary is the condition on all results of the chapter).

References

[1] H. Barringer and D. Rydeheard. A note on logical description, observational orders and minimum models. Available online at: http://www.cs.man.ac.uk/ david/evolution/logic.pdf, 2008.

[2] G. S. Boolos, J. P. Burgess, and R. C. Jeffrey. *Computability and Logic.* Cambridge University Press, 2007.