

Logic, Computability and Incompleteness

The Halting Problem and
Undecidability

Decidability

We have seen that the halting function h is **not Turing Computable**

and that if the **Church-Turing Thesis** is true, then the **halting problem** is **absolutely unsolvable**.

- We will now extend these negative, limitative results to the realm of **First-Order Logic** (FOL) and show that FOL is **undecidable**.
- In general, given some set of objects of the relevant sort, a property \mathcal{P} of such objects is **decidable** just in case there is an effective, mechanical procedure such that, for any arbitrary object of the relevant sort,

Decidability

the procedure eventually classifies the object (*correctly!*) as a positive or negative instance of that property.

- More specifically,
 - (i) if a given object *has* the property \mathcal{P} , then the procedure determines in finitely many steps that *it has* the property,
 - (ii) if the object *does not have* the property \mathcal{P} , then the procedure determines in finitely many steps that *it does not have* the property.
- The property \mathcal{P} is *decidable* iff we have **both** (i) and (ii).

Decidability

For example, let Ψ be an arbitrary (finite) string of symbols from the primitive vocabulary of our formal object language L for FOL.

Now consider the property of

being a grammatically well formed formula of L .

It turns out (unsurprisingly!) that this property is **decidable**.

So, if Ψ is some arbitrary (finite) expression, we can decide in finitely many steps whether or not Ψ is a formula.

Logical Validity

- In the present context, objects of the relevant sort are **closed formulas** (i.e. sentences) of the language L of FOL and the property of interest is **satisfiability/logical validity**.
- We will show that there is no mechanical positive test for FOL **satisfiability**, and hence, equivalently, for FOL **invalidity**.
- Essential connection between **validity** and **satisfiability**:
 $\Gamma \models \Psi$ iff the set $\Gamma \cup \{\neg \Psi\}$ is **unsatisfiable**.

If the set $\Gamma \cup \{\neg \Psi\}$ *did* have a model then it would be a counter-model to the claim $\Gamma \models \Psi$.

Thus the **satisfiability** of $\Gamma \cup \{\neg \Psi\}$ is tantamount to **invalidity** and means that $\Gamma \not\models \Psi$.

Logical Validity

- We will soon prove that FOL is complete, since there is an effective positive method for determining validity.
- But this is only clause (i) of the decidability problem, and as we will now show, clause (ii) fails, since there is no effective method for determining the negative cases of invalidity.
- Hence validity overall in FOL is undecidable.
- We'll use a reductio argument to show that there can be no effective method for determining cases of invalidity, by showing that if there were such a test, then the halting problem for Turing Machines would be solvable (and hence the Church-Turing Thesis would be false)

Logical Validity and the Halting Problem

- The strategy is analogous to the demonstration that Turing computable functions are Recursive.
- There, we constructed recursive functions which **numerically replicate** the sequence of configurations of any given TM computation.
- Here, we'll construct a finite set of **FOL sentences** which exactly characterize or **formalize** the sequence of configurations.
- We'll specify a method such that, given any TM and any input n , we can construct a finite set of sentences Δ and a corresponding sentence **H** such that

$\Delta \models \mathbf{H}$ iff the TM eventually halts on input n (!)

Logical Validity and the Halting Problem

- Thus **validity** in FOL is directly correlated with **halting**, and **invalidity** with **non-halting**.
- So **if** validity in FOL were decidable **then** we could solve the halting problem and compute ***h***, as follows:
- Given the m^{th} TM, say TM_m , and a selected input n , we construct the set of sentences Δ .
Given the specification of TM_m we also construct ***H***.
- Then, if $\Delta \models \textbf{\textcolor{red}{H}}$ then ***h*** (m,n) = 2 and
 if $\Delta \not\models \textbf{\textcolor{red}{H}}$ then ***h*** (m,n) = 1
- We could thereby compute the (provably) non-Turing computable function ***u*** as well.

Formalizing TM Computations in FOL

- To begin the construction, let all the squares of the machine tape be numbered, with 0 as the starting square of the computation:

... | -2 | -1 | 0 | 1 | 2 | 3 | ... [this number is distinct from the symbol occurring in the square]

- Let t be the ‘time’ variable ranging over steps in the computation
- FOL Vocabulary:

For each **state** q_i of machine \mathbf{M} , pick a 2-place predicate \mathbf{Q}_i

For each **symbol** S_j the machine can read/write, pick a 2-place predicate \mathbf{S}_j

Formalizing TM Computations in FOL

- Also need \mathbf{o} (a selected individual constant), $'$ (a 1-place function), $<$ (a 2-place relation), and $=$
- **Intended interpretation \mathcal{J}** : $D =$ set of integers
 $tQ_i x$ is true **iff** at time t \mathbf{M} is in state q_i scanning square number x
 $tS_j x$ is true **iff** at time t the symbol S_j is in square number x .
 The interpretations of $\mathbf{o}, ', <$ are standard.
- With the **intended interpretation \mathcal{J}** in mind, and the supposition that it can read/write only the symbols S_0, \dots, S_r , will now **axiomatize** a given machine's operation as follows.
 There are only 3 types of **instruction** in \mathbf{M} 's specification:

Formalizing TM Computations in FOL

- $q_i S_j S_k q_m$

Corresponding axiom

$$\forall t \forall x \forall y[(tQ_i x \wedge tS_j x) \rightarrow (t'Q_m x \wedge t'S_k x \wedge (y \neq x \rightarrow (tS_0 y \rightarrow t'S_0 y \wedge \dots \wedge tS_r y \rightarrow t'S_r y)))]$$

Under the **intended interpretation**, this axiom says:

If machine **M** is in state q_i at time t scanning square x on which the symbol S_j occurs, then at time $t+1$ **M** is in state q_m scanning square x where the symbol S_k occurs, and in all squares other than x , the same symbols appear at time $t+1$ as at time t .

Formalizing TM Computations in FOL

- $q_i S_j R q_m$

axiom:

$$\forall t \forall x \forall y [(tQ_i x \wedge tS_j x) \rightarrow \\ (t'Q_m x' \wedge (tS_0 y \rightarrow t'S_0 y \wedge \dots \wedge tS_r y \rightarrow t'S_r y))]$$

- $q_i S_j L q_m$

axiom:

$$\forall t \forall x \forall y [(tQ_i x' \wedge tS_j x') \rightarrow \\ (t'Q_m x \wedge (tS_0 y \rightarrow t'S_0 y \wedge \dots \wedge tS_r y \rightarrow t'S_r y))]$$

Add all such axioms to Δ according to the set of quadruples defining **M**.

Formalizing TM Computations in FOL

- Now formalize the initial configuration for input n
(where $t = 0$, $x = 0$ and **state** = q_1)

$$\mathbf{oQ}_1\mathbf{o} \wedge \mathbf{oS}_1\mathbf{o} \wedge \mathbf{oS}_1\mathbf{o}' \wedge \dots \wedge \mathbf{oS}_1\mathbf{o}^{(n-1)} \wedge$$

$$\forall y ((y \neq \mathbf{o} \wedge y \neq \mathbf{o}' \wedge \dots \wedge y \neq \mathbf{o}^{(n-1)}) \rightarrow \mathbf{oS}_0y))$$

- Finally need to throw in two arithmetical axioms to govern behavior of $'$ and $<$
- First axiom says that each integer is the successor of exactly one integer: $\forall z \exists x (z = x') \wedge \forall z \forall x \forall y ((z = x' \wedge z = y') \rightarrow x = y)$
- Second axiom:

$$\forall x \forall y \forall z (x < y \wedge y < z \rightarrow x < z) \wedge \forall x \forall y (x' = y \rightarrow x < y) \\ \wedge \forall x \forall y (x < y \rightarrow x \neq y)$$

Formalizing TM Computations in FOL

- This yields the set Δ formalizing the machine \mathbf{M} started in initial configuration on input n .
- Next need to define H . Note that a machine halts at time t iff it is in state q_i reading a symbol S_j and there is no quadruple beginning with the pair $q_i S_j$.
- There will be a **finite** number of such possibilities, and the pairs for any given machine can be determined by inspection.
- For each such pair $q_i S_j$, construct the sentence
$$\exists t \exists x (tQ_i x \wedge tS_j x)$$

And let H be the disjunction of all such existential sentences for a given machine \mathbf{M} .

$\Delta \models H$ iff \mathbf{M} halts on input n

- If there are no such halting pairs, then \mathbf{M} never halts, and so let H be $0 \neq 0$.
- The construction is now complete!
- Claim: Given machine \mathbf{M} and input n , we have engineered Δ and H such that

$\Delta \models H$ iff \mathbf{M} eventually halts on input n

- Verification of claim: the ‘only if’ direction is trivial:
it’s clear that if $\Delta \models H$, then \mathbf{M} eventually halts on input n ,
since the intended interpretation \mathcal{J} is a model of Δ ,
and H is true in \mathcal{J} iff \mathbf{M} halts.

if \mathbf{M} halts on n , *then* $\Delta \models H$

- The harder part of the **verification** is the ‘*if*’ direction, i.e. to show that *if* \mathbf{M} halts on n , *then* $\Delta \models H$.
- Recall that a TM computation is a sequence of configurations, where a **configuration** includes the **current state**, **scanned square** on the tape, and the **contents** of all non-blank squares on the tape.
- So *a description of time s* is a characterization of the configuration at stage s in the computation.
- A *description of time s* will be a conjunction of the form:

$$\mathbf{o}^{(s)}\mathbf{Q}_i\mathbf{o}^{(p)} \wedge \mathbf{o}^{(s)}\mathbf{S}_{j1}\mathbf{o}^{(p1)} \wedge \dots \wedge \mathbf{o}^{(s)}\mathbf{S}_{j}\mathbf{o}^{(p)} \wedge \dots \wedge \mathbf{o}^{(s)}\mathbf{S}_{jv}\mathbf{o}^{(pv)} \wedge$$

$$\forall y [(y \neq \mathbf{o}^{(p1)} \wedge y \neq \mathbf{o}^{(p)} \wedge \dots \wedge y \neq \mathbf{o}^{(pv)}) \rightarrow \mathbf{o}^{(s)}\mathbf{S}_0y]$$

What does it mean?

$$\mathbf{o}^{(s)}\mathbf{Q}_{\textcolor{red}{i}}\mathbf{o}^{(\textcolor{violet}{p})} \wedge \mathbf{o}^{(s)}\mathbf{S}_{j1}\mathbf{o}^{(p1)} \wedge \dots \wedge \mathbf{o}^{(s)}\mathbf{S}_{\textcolor{red}{j}}\mathbf{o}^{(\textcolor{violet}{p})} \wedge \dots \wedge \mathbf{o}^{(s)}\mathbf{S}_{jv}\mathbf{o}^{(pv)} \wedge \\ \forall y \left[(y \neq \mathbf{o}^{(p1)} \wedge y \neq \mathbf{o}^{(\textcolor{violet}{p})} \wedge \dots \wedge y \neq \mathbf{o}^{(pv)}) \rightarrow \mathbf{o}^{(s)}\mathbf{S}_0y \right]$$

1st conjunct $\mathbf{o}^{(s)}\mathbf{Q}_{\textcolor{red}{i}}\mathbf{o}^{(\textcolor{violet}{p})}$: current state and scanned square at time s

The next string of conjuncts

$$\mathbf{o}^{(s)}\mathbf{S}_{j1}\mathbf{o}^{(p1)} \wedge \dots \wedge \mathbf{o}^{(s)}\mathbf{S}_{\textcolor{red}{j}}\mathbf{o}^{(\textcolor{violet}{p})} \wedge \dots \wedge \mathbf{o}^{(s)}\mathbf{S}_{jv}\mathbf{o}^{(pv)}$$

gives the non-empty tape contents. This occurs in the range squares $p_1, \dots, \textcolor{violet}{p}, \dots, p_v$, where the central term $\mathbf{o}^{(s)}\mathbf{S}_{\textcolor{red}{j}}\mathbf{o}^{(\textcolor{violet}{p})}$ indicates the symbol in the currently scanned square.

Finally $\forall y \left[(y \neq \mathbf{o}^{(p1)} \wedge y \neq \mathbf{o}^{(\textcolor{violet}{p})} \wedge \dots \wedge y \neq \mathbf{o}^{(pv)}) \rightarrow \mathbf{o}^{(s)}\mathbf{S}_0y \right]$ states that the rest of the tape is blank.

if \mathbf{M} halts on n , *then* $\Delta \models H$

- Now suppose that \mathbf{M} eventually halts on input n .
Then for some s , i , p and j , it is the case that at time s the machine is in state q_i scanning square number p in which the symbol S_j appears,
and there is no quadruple in its program beginning $q_i S_j$.
- Suppose further that Δ implies a description D of time s .
Since \mathcal{J} is a model of Δ , D will be true in \mathcal{J} .
Two of the *conjuncts* of D will be $\mathbf{o}^{(s)}\mathbf{Q}_i \mathbf{o}^{(p)}$ and $\mathbf{o}^{(s)}\mathbf{S}_j \mathbf{o}^{(p)}$
and therefore D will imply $\exists t \exists x (t\mathbf{Q}_i x \wedge t\mathbf{S}_j x)$
which is one of the *disjuncts* of H .
- Therefore $\Delta \models H$

Proof by Mathematical Induction

- Hence we only need to show that for each s , if M has not halted before time s , then Δ implies a description of time s .
- Will prove this through Mathematical Induction.
- The basic idea of a proof by **mathematical induction** is to show that some statement or property holds for all possible cases (or values of n).

The proof consists of two steps:

The **basis step**: prove that the statement or property holds for the minimal case (or first ‘natural number’ n , so usually, $n = 0$ or $n = 1$).

Proof by Mathematical Induction

The **induction step**: prove that, *if* the statement holds for all cases up to (or at) an arbitrary given point (some natural number n), *then* the statement or property holds for all cases at the next higher point (or $n + 1$).

The hypothesis in the induction step that the statement holds for some arbitrary given point n is called the **induction hypothesis**.

To perform the **induction step**, assume the induction hypothesis is true, and then use this assumption to prove that the statement holds at the next level ($n + 1$).

Proof by Mathematical Induction

Basis step: $s = 0$. The sentence

$$\mathbf{oQ}_1\mathbf{o} \wedge \mathbf{oS}_1\mathbf{o} \wedge \mathbf{oS}_1\mathbf{o}' \wedge \dots \wedge \mathbf{oS}_1\mathbf{o}^{(n-1)} \wedge \\ \forall y ((y \neq \mathbf{o} \wedge y \neq \mathbf{o}' \wedge \dots \wedge y \neq \mathbf{o}^{(n-1)}) \rightarrow \mathbf{oS}_0y))$$

is a description of time 0 and is contained in Δ .

Induction step: our induction hypothesis is the statement:

if \mathbf{M} has not halted before time s ,

then Δ implies *a description of time s* . Suppose this is true for s .

For **induction step**, assume the antecedent, that

\mathbf{M} has not halted before time $s + 1$.

Must then show the consequent, that

Δ implies a description of time $s + 1$.

Proof by Mathematical Induction

Since \mathcal{J} is a model of Δ , the *description of time s* is true in \mathcal{J} :

$$\mathbf{o}^{(s)}\mathbf{Q}_i\mathbf{o}^{(p)} \wedge \mathbf{o}^{(s)}\mathbf{S}_{j1}\mathbf{o}^{(p1)} \wedge \dots \wedge \mathbf{o}^{(s)}\mathbf{S}_j\mathbf{o}^{(p)} \wedge \dots \wedge \mathbf{o}^{(s)}\mathbf{S}_{jv}\mathbf{o}^{(pv)} \wedge \\ \forall y ((y \neq \mathbf{o}^{(p1)} \wedge y \neq \mathbf{o}^{(p)} \wedge \dots \wedge y \neq \mathbf{o}^{(pv)}) \rightarrow \mathbf{o}^{(s)}\mathbf{S}_0y))$$

- So at time s \mathbf{M} is in state q_i scanning square number p in which the symbol S_j appears.
- Since \mathbf{M} does not halt at time s , its program must contain a quadruple of *exactly* one of the following 3 forms:
 - (i) $q_i S_j S_k q_m$
 - (ii) $q_i S_j R q_m$
 - (iii) $q_i S_j L q_m$

Proof by Mathematical Induction

If (i) $q_i S_j S_k q_m$ then one of the sentences in Δ is

$$\forall t \forall x \forall y [(tQ_i x \wedge tS_j x) \rightarrow (t'Q_m x \wedge t'S_k x \wedge (y \neq x \rightarrow (tS_0 y \rightarrow t'S_0 y \wedge \dots \wedge tS_r y \rightarrow t'S_r y)))]$$

This, together with the *description of time s*

(and the 2 axioms for ', <) *imply the sentence*

$$\mathbf{o}^{(s+1)}Q_m \mathbf{o}^{(p)} \wedge \mathbf{o}^{(s+1)}S_{j1} \mathbf{o}^{(p1)} \wedge \dots \wedge \mathbf{o}^{(s+1)}S_k \mathbf{o}^{(p)} \wedge \dots \wedge \mathbf{o}^{(s+1)}S_{jv} \mathbf{o}^{(pv)} \wedge \\ \forall y ((y \neq \mathbf{o}^{(p1)} \wedge y \neq \mathbf{o}^{(p)} \wedge \dots \wedge y \neq \mathbf{o}^{(pv)}) \rightarrow \mathbf{o}^{(s+1)}S_0 y))$$

which is a *description of time s + 1*

Proof by Mathematical Induction

If (ii) $q_i S_j R q_m$ then one of the sentences in Δ is

$$\forall t \forall x \forall y [(tQ_i x \wedge tS_j x) \rightarrow (t'Q_m x' \wedge (tS_0 y \rightarrow t'S_0 y \wedge \dots \wedge tS_r y \rightarrow t'S_r y))]$$

There is some symbol S_g such that the above together with the *description of time s* (and the 2 axioms for ', <)

imply the sentence

$$o^{(s+1)}Q_m o^{(p+1)} \wedge o^{(s+1)}S_{j1} o^{(p1)} \wedge \dots \wedge o^{(s+1)}S_j o^{(p)} \wedge o^{(s+1)}S_g o^{(p+1)} \\ \wedge \dots \wedge o^{(s+1)}S_{jv} o^{(pv)} \wedge$$

$$\forall y ((y \neq o^{(p1)} \wedge y \neq o^{(p)} \wedge y \neq o^{(p+1)} \dots \wedge y \neq o^{(pv)}) \rightarrow o^{(s+1)}S_0 y))$$

which is a *description of time s + 1*

Proof by Mathematical Induction

If (iii) $q_i S_j L q_m$ then one of the sentences in Δ is

$$\forall t \forall x \forall y [(tQ_i x' \wedge tS_j x') \rightarrow (t'Q_m x \wedge (tS_0 y \rightarrow t'S_0 y \wedge \dots \wedge tS_r y \rightarrow t'S_r y))]$$

There is some symbol S_g such that the above together with the *description of time s* (and the axioms for $'$, $<$) imply the sentence

$$\mathbf{o}^{(s+1)}Q_m \mathbf{o}^{(p-1)} \wedge \mathbf{o}^{(s+1)}S_{j1} \mathbf{o}^{(p1)} \wedge \dots \wedge \mathbf{o}^{(s+1)}S_g \mathbf{o}^{(p-1)} \wedge \mathbf{o}^{(s+1)}S_j \mathbf{o}^{(p)} \wedge \dots \wedge \mathbf{o}^{(s+1)}S_{jv} \mathbf{o}^{(pv)} \wedge$$

$$\forall y ((y \neq \mathbf{o}^{(p1)} \wedge y \neq \mathbf{o}^{(p-1)} \wedge y \neq \mathbf{o}^{(p)} \dots \wedge y \neq \mathbf{o}^{(pv)}) \rightarrow \mathbf{o}^{(s+1)}S_0 y))$$

which is a *description of time $s + 1$* .

In all 3 cases Δ implies a description of time $s + 1$. \square

Hence $\Delta \models H$ iff \mathbf{M} halts on input n .

FOL is Undecidable

And since we have just proved the non-trivial direction of the biconditional, viz. *if* \mathbf{M} halts on n , *then* $\Delta \models H$

contraposition yields

if $\Delta \not\models H$ then \mathbf{M}_m does **not** halt on input n ,

and $h(m, n) = 1$

So if **FOL** were decidable, then there would be an effective procedure for determining that $\Delta \not\models H$

So if **FOL** were decidable then the halting function would be computable (and the Church-Turing Thesis would be refuted).

Hence it follows by our original *reductio* strategy that

FOL is **undecidable**.