Computability (Algorithms) for propositional logic

Encode formulas or finite sets of formulas over a finite alphabet:

$$\Sigma^{prop} = \{ \wedge, \vee, \neg, \rightarrow, \leftrightarrow, (,), p, 0, 1, \ldots, 9 \}$$

Example $p_1 \vee \neg p_{17}$ is encoded by $(p1 \vee \neg p17) \in (\Sigma^{prop})^*$

For sets of formulas $\Sigma^{prop+} := \Sigma^{prop} \cup \{,\}$ comma symbol

$\{p_1, p_1 \vee p_{17}\}$ encoded by $p1, (p1 \vee p17) \in (\Sigma^{prop+})^*$

Thm The set of syntactically valid propositional formulas is decidable.

Here we are conflating formulas and the strings in $(\Sigma^{prop})^*$ that represent them.

Thm The set of tautologies is decidable.

Pf Use method of truth tables. $\boxed{\checkmark}$

Thm The set of finite sets Γ of syntactically correct formulas is decidable.

Thm • The binary relation $\langle \Pi, A \rangle$ s.t. Π is a finite set of formulas, A is a formula and $\Pi \models A$ is decidable.

• The set of satisfiable set Π of sentences is decidable

• " " " contradictory sets " " " " " "

Pf Method of truth tables.

Thm: Suppose Π is a c.e. set of formulas then, the $\{A : \Pi \models A\}$ is c.e.

Pf: Algorithm that semidecides this set is

Assumption: M enumerates Π

Input A $\quad (w \in (\Sigma^{prop})^*)$

Algorithm

If w does not correctly encode a formula, reject

For $i = 0, 1, 2, 3, \ldots$

- Run M until it enumerates i member of Π
- Let Γ_i be those first i formulas
- If $\Gamma_i \models A$, accept (and halt)

End-for

Algorithm semidecides $\{A : \Pi \models A\}$ by the Compactness theorem. $\square$

Above theorem ~~a~~ cannot be strengthened for <u>decidable</u> Π to conclude that $\{A : \Pi \models A\}$ is <u>decidable</u>.

<u>Example</u> Let A be p_1

Suppose Π <u>might</u> contain one of p_1 or

$p_1 \wedge p_1$ or $p_1 \wedge p_1 \wedge p_1$ or

$\bigwedge_{j=1}^{k} p_1$

<u>Exercise</u> If Π is c.e. there is a decidable T such $T \models \# \Pi$ (Craig's Theorem)

Algorithm for first-order logic

Encode first-order formulas on alphabet (for L a finite language)

$$\Sigma^{fo\text{-}L} = \{\rightarrow, \neg, \forall, (,), x, 0, \ldots, 9, =\} \cup L$$

$$\forall x_1 (x_1 = f(x_1)) \qquad \forall x1\ x1 = f(x1)$$

Encode finite sets of first-order formulas using

$$\Sigma^{fo\text{-}L+} = \Sigma^{fo\text{-}L} \cup \{,\}.$$

<u>Thm</u> Set of syntactically correct L-formulas is decidable

Encode <u>proofs in FO</u>, by a sequence of formulas separated by commas as $(\Sigma^{fo\text{-}L+})$-string

<u>Thm</u>: The binary relation $\langle \Pi, P \rangle$ of pairs Π and P s.t. P is a valid FO proof from the hypotheses Π is decidable.

<u>Thm</u>: the set of valid L-formula is c.e.

<u>Pf</u> A is valid iff A has a proof.

<u>Idea</u>: Search for a ~~proof~~ proof A. Accept if a proof is found.

<u>Input</u> A

<u>Algorithm</u>

For $i = 1, 2, 3, \ldots$

Let v_i be the i-th member of $(\Sigma^{fo\text{-}L+})^*$

If v_i encodes a valid proof P of A, accept (and halt)

End-for

Thm The set of pairs $\langle \Pi, A \rangle$ s.t. Π is a finite set of formulas, A is a formula, and $\Pi \models A$, is ~~decidable~~ computably enumerable.

Pf: Algorithm

Enumerate all pairs $\langle \Gamma, P \rangle$ s.t. P is a valid proof from by hypotheses Γ. If $\Pi = \Gamma$ and A is the final formula in P, accept.

Theorem Let Γ be a c.e. set of formulas. Then $\{A : \Gamma \models A,\ A \text{ is a formula}\}$ is c.e.

Proof Algorithm idea: Start enumerating Γ and all possible proofs (dove tailing) + watch for a proof of A from a finite subset of Γ.

Assumption: M enumerates Γ

Input A

Algorithm For $i = 1, 2, 3, \ldots$

Run M for i steps, let Γ_i be the subset of Γ enumerated by M within i steps

For $j = 1, 2, 3, \ldots$

Let u_j be the j-th member of $(\Sigma^{\text{fo-lit}})^*$

If u_j encodes a valid proof of A from Γ_i, accept

End for

End for

<u>Theorem</u> If T is a complete, c.e. theory, then T is decidable

(In the PDF)