Topics

(1) Undecidable Theories

(2) Gödel's incompleteness theorem (1st)

(3) Computation in HA

(4) Constructive Issues

- Markov's Principle
  - in iFOL
  - in HA
- Limited computations
- Next week Constructive Type Theory

1 Undecidable Theories

Recall, a set of numbers S is *definable* in a theory T iff there is a predicate F_S in T such that

$$n \in S \text{ implies } \models_T F_S(\overline{n}) \qquad \text{and}$$
$$n \notin S \text{ implies } \models_T \sim F_S(\overline{n})$$

Lemma: If T is a consistent extension of $\mathcal{Q}$, then $TrueT(x)$ the Gödel numbers of true formulas of T is not definable in T.

Proof: Suppose $TrueT(x)$ is definable, then by the Diagonal Lemma there is a formula G of T such that

$$\models_T G \Leftrightarrow \sim TrueT(\ulcorner G \urcorner)$$

then we can prove $\models_T G$ and $\models_T \sim G$, so T is not consistent.

Qed

Definition: A theory T is *recursively decidable* iff $TrueT(x)$ is recursive.

A theory T is *decidable* iff $\forall x : \mathbb{N}.\big(TrueT(x) \vee \sim TrueT(x)\big)$ is constructively true.

Theorem 1: No consistent extension of $\mathcal{Q}$ is recursively decidable.

Proof: If T is a consistent extension of $\mathcal{Q}$ and T is recursively decidable, then some recursive function d decides it, say

$$d(x) = 1 \text{ iff } \mathit{TrueT}(x)$$

But since the function d is representable in $\mathcal{Q}$, there is a predicate $D(x, y)$ that represents it.

Thus $D(x, 1)$ defines $\mathit{TrueT}(x)$ in T. But there is no such formula by the above Lemma.

Qed

Corollary: $i\mathcal{Q}$ is not decidable recursively.

Theorem 2: (Church) FOL is not decidable.

Proof: $\mathcal{Q}$ is finitely axiomatizable in FOL, so if $\mathcal{Q}Ax$ is a finite conjunction of the axioms of $\mathcal{Q}$, then

$$\models_{\mathcal{Q}} A \text{ iff } \models_{FOL} (\mathcal{Q}Ax \Rightarrow A).$$

So if FOL is decidable, then so is $\mathcal{Q}$.

Qed

Note, using the completeness theorem for iFOL proved by Mark Bickford and me we know

$$\models_{i\mathcal{Q}} A \text{ iff } \models_{iFOL} (\mathcal{Q}Ax \Rightarrow A).$$

This is a basis for extending these results above to $i\mathcal{Q}$ and its extensions. We will not do this work in the course.

Now we look at computation in HA. Consider why we know that $\forall x, y : \mathbb{N}.(x = y \lor x \neq y)$. This theorem says that we can decide whether $n = m$ for two numbers. How would we do this?

$eq(n, m) ==$ if $n = 0$ then ask if $m = 0$ *then* true *else* false
else if $m = 0$ *then* false *else* $(n - 1 = m - 1)$ $eq(n - 1, m - 1)$

This is a fairly good algorithm, but it is NOT primitive recursive, so it does not come directly from an inductive proof. How do we prove this in HA?

See lecture for topics (2)-(4).