CHAPTER 14

Systems for arithmetic

1. Gödel's $\mathcal{T}$

In an earlier chapter we introduced the typed lambda calculus over a set of base types A. In this chapter on arithmetic we will consider this system again, but we will only work with one base type, which we will think of as the set of natural numbers $\mathbb{N}$ and which is traditionally denoted by 0. We will also have three new constants and two new reduction rules and the result will be a rewriting system which is called Gödel's $\mathcal{T}$.

DEFINITION 1.1. The finite types are defined by induction as follows: 0 is a finite type, and if σ and τ are finite types, then so are $\sigma \to \tau$ and $\sigma \times \tau$.

Besides the combinators $\mathbf{p}$, $\mathbf{p_0}$ and $\mathbf{p}_i$, Gödel's $\mathcal{T}$ includes the following constants: a constant 0 of type 0, which stands for the natural number zero, a constant S of type $0 \to 0$ which stands for the successor function and for each type σ a "recursor" $\mathbf{R}^\sigma$ of type $\sigma \to ((0 \to (\sigma \to \sigma)) \to (0 \to \sigma))$, whose meaning will be explained shortly. So the constants (combinators) of Gödel's $\mathcal{T}$ are:

(1) for each pair of types ρ, σ combinators $\mathbf{p}^{\rho,\sigma}, \mathbf{p}_0^{\rho,\sigma}, \mathbf{p}_1^{\rho,\sigma}$ of types $\rho \to (\sigma \to \rho \times \sigma)$, $\rho \times \sigma \to \rho$ and $\rho \times \sigma \to \sigma$, respectively.
(2) a constant 0 of type 0 and a constant S of type $0 \to 0$.
(3) for each type σ a combinator $\mathbf{R}^\sigma$ ("the recursor") of type $\sigma \to ((0 \to (\sigma \to \sigma)) \to (0 \to \sigma))$.

The terms of Gödel's $\mathcal{T}$ are built as before:

DEFINITION 1.2. The terms of a certain type are defined inductively as follows:

- each variable or constant of type σ will be a term of type σ.
- if s is a term of type $\sigma \to \tau$ and t is a term of type σ, then st is a term of type τ.
- if x^σ is a variable of type σ and t is a term of type τ, then $\lambda x^\sigma.t$ is a term of type $\sigma \to \tau$.

Besides the old reduction rules there will be two new ones.

DEFINITION 1.3. An expression on the left of the table below is called a *redex*. If t is a redex and t' is the corresponding expression on the right of the table, then we will say that t *converts to* t' and we will write t conv t'.

t	t'
$(\lambda x.s)t$	$s[t/x]$
$\mathbf{p}_i(\mathbf{p}t_0t_1)$	t_i
$\mathbf{R}t_1t_20$	t_1
$\mathbf{R}t_1t_2(Sn)$	$t_2n(\mathbf{R}t_1t_2n)$

The definition of the reduction relation, a reduction sequence, a normal form *et cetera* are as before.

The final two reduction rules tell us how we should think of the recursor. The recursor of type σ yields for every pair of values t_1 of type σ and t_2 of type $0 \to (\sigma \to \sigma)$ a sequence of objects of type σ, defined by recursion. That is, $\mathbf{R}t_1t_2$ is a function of type $0 \to \sigma$, so expects a natural number m as input. If $m = 0$, then $\mathbf{R}t_1t_2m$ is t_1, which is the starting point of the recursion. If $m = Sn$, then we may assume that $\mathbf{R}t_1t_2n$ has already been defined and the recursion should tell us how the new value $\mathbf{R}t_1t_2(Sn)$ can be obtained from the value $\mathbf{R}t_1t_2n$ and the natural number n. In fact, that is where t_2 comes in, in telling us how the next value is to be computed from the previous: $\mathbf{R}t_1t_2(Sn)$ is given by feeding t_2 with n and the previous value Rt_1t_2n.

As for the typed lambda calculus, we have that the rewriting rules for Gödel's $\mathcal{T}$ make it both strongly normalising and confluent.

THEOREM 1.4. (Strong normalisation for Gödel's $\mathcal{T}$, Tait, 1967) *The rewriting system given by Gödel's $\mathcal{T}$ is strongly normalising and confluent. That is, for any term t in Gödel's $\mathcal{T}$ there is a number n such that every reduction sequence starting from t has length at most n and ends with the same term in normal form.*

PROOF. The proof is an adaptation of the strong normalisation proof for the typed lambda calculus. For the student who is interested: in the definition of a neutral expression the ones which now also have to be excluded are 0 and St. With these modications we can adapt the proof of Lemma 1.12 from Chapter 8. Also, one has to prove that $\mathbf{R}t_1t_2t_3$ is computable whenever the t_i are, which one does by induction on $\nu(t_1) + \nu(t_2) + \nu(t_3) + m$, where m is the number of symbols in the normal form of t_3. □

The result is a system in which one can do quite a bit of arithmetic. For example, there is a constant **plus** of type $0 \to (0 \to 0)$ such that:

$$\begin{aligned} \mathbf{plus}\,0\,n &\succeq n \\ \mathbf{plus}\,(Sm)\,n &\succeq S(\mathbf{plus}\,m\,n) \end{aligned}$$

Indeed, one may define **plus** as

$$\mathbf{plus} := \mathbf{R}^{0\to 0}(\lambda x^0.x)(\lambda y^0.\lambda f^{0\to 0}.\lambda x^0.S(fx))$$

(please check!).

For example, we have:

$$\mathbf{plus}\,(SS0)\,(SS0) \succeq S[\mathbf{plus}\,(S0)\,(SS0)] \succeq SS[\mathbf{plus}\,0\,(SS0)] \succeq SSSS0.$$

So $2 + 2 = 4$, which you probably already knew.

In a similar way one can define terms for multiplication and exponentiation, for instance. Since the system is strongly normalising and normal forms are unique, and because the only closed terms in normal form of type 0 are numerals S^m0, one can really do arithmetic inside Gödel's $\mathcal{T}$.

2. Arithmetic in all finite types

In this section we combine Gödel's $\mathcal{T}$ with first-order logic to give us a system in which one can also prove some arithmetical theorems.

The system HA^ω is formulated in multi-sorted intuitionistic logic, with the sorts being the finite types and the terms being those of Gödel's $\mathcal{T}$.

DEFINITION 2.1. The formulas of HA^ω are defined inductively as follows:

- $\bot$ is a formula and if s and t are terms of Gödel's $\mathcal{T}$ of the same type σ, then $s =_\sigma t$ is a formula.
- if φ and ψ are formulas, then so are $\varphi \wedge \psi$ and $\varphi \to \psi$.
- if x is a variable of type σ and φ is a formula, then $\exists x^\sigma\, \varphi$ and $\forall x^\sigma\, \varphi$ are formulas.

The axioms and rules of HA^ω are:

(i) All the axioms and rules of intuitionistic logic (say in Hilbert-style).
(ii) Rules for equality at all types:

$$x =_\sigma x, \qquad x =_\sigma y \to y =_\sigma x, \qquad x =_\sigma y \wedge y =_\sigma z \to x =_\sigma z,$$

$$f =_{\sigma\to\tau} f' \wedge x =_\sigma x' \to fx =_\tau f'x'.$$

(iii) The successor axioms:

$$\neg S(x) =_0 0, \qquad S(x) =_0 S(y) \to x =_0 y$$

(iv) For any formula φ in the language of HA^ω, the induction axiom:

$$\varphi(0) \to \big(\forall x^0\,(\,\varphi(x) \to \varphi(Sx)\,) \to \forall x^0\,\varphi(x)\,\big).$$

(v) The axioms for λ and the combinators:

$$\begin{aligned} (\lambda x^\sigma.s)t &= s[t/x] \\ \mathbf{p}_0(\mathbf{p}xy) &= x \\ \mathbf{p}_1(\mathbf{p}xy) &= y \end{aligned}$$

as well as for the recursor:

$$\begin{aligned} \mathbf{R}xy0 &= x \\ \mathbf{R}xy(Sn) &= yn(\mathbf{R}xyn) \end{aligned}$$

In HA^ω we cannot prove the following *extensionality axioms*:

$$\forall f^{\sigma\to\tau}, g^{\sigma\to\tau}\big(\,(\forall x^\sigma fx =_\tau gx) \to f =_{\sigma\to\tau} g\big)$$
$$\forall x^{\sigma\times\tau}, y^{\sigma\times\tau}\,\big(\,\mathbf{p}_0x =_\sigma \mathbf{p}_0y \wedge \mathbf{p}_1x =_\tau \mathbf{p}_1y \to x =_{\sigma\times\tau} y\,\big)$$

The result of adding these axioms to HA^ω will be denoted by $\mathsf{E\text{-}HA}^\omega$.

To both HA^ω and $\mathsf{E\text{-}HA}^\omega$ we can add the Law of Excluded Middle $\varphi \vee \neg\varphi$ or Double Negation Elimination $\neg\neg\varphi \to \varphi$: we will denote the resulting systems by PA^ω and $\mathsf{E\text{-}PA}^\omega$, respectively.

Some remarks about these systems:

(1) All these systems satisfy the deduction theorem, so one can freely use natural deduction to prove things in these systems.
(2) For any formula $\varphi(x)$ in the language of HA^ω we have

$$\mathsf{HA}^\omega \vdash x =_\sigma y \wedge \varphi(x) \to \varphi(y).$$

Indeed, this is quite easy to prove by induction on the structure of φ. And from this it follows that the same is provable in all other systems, because HA^ω is a subsystem of all of them.
(3) If $t \succeq t'$ in Gödel's $\mathcal{T}$, then $\mathsf{HA}^\omega \vdash t = t'$.

(4) The alert reader will have noticed that we have not included disjunction in the syntax of HA^ω. The reason is that we will regard disjunction as a defined symbol, as follows:

$$\varphi \vee \psi :\equiv \exists n^0 \left[(n = 0 \to \varphi) \wedge (\neg n = 0 \to \psi) \right].$$

One should check that the usual logical axioms for disjunction can now be proved on the basis of the other axioms of HA^ω.

In HA^ω one can now prove things like: addition is associative and commutative. Please try!