2 Heyting Arithmetic

2.1 Equality

The theories we will consider will also have equality. For this we add a relation symbol $=^S$ of arity S, S for every sort S. We then add the following axioms whenever x is a free variable of sort S and s and t are terms of sort S. The axiom scheme is only for *atomic* formulas φ, i.e. those of the form $Rt_1 \dots t_n$ for a relation symbol R, but holds without loss of generality for all formulas.

$$\Gamma \vdash x =^S x \qquad \frac{\Gamma \vdash s = t \qquad \Gamma \vdash \varphi[x/s]}{\Gamma \vdash \varphi[x/t]}$$

2.2 Review of first order Heyting arithmetic

The first formal theory we will see is (first order) Heyting arithmetic, **HA**. This is one of the most basic theories where we can formalise some real parts of mathematics. The signature for **HA** has a single sort, which we think of as the set of natural numbers. It has a nullary operator 0, a unary operator S, two binary operators $+$ and $\times$ and equality. We use *infix* notation for $+$ and $\times$. That is, we write $n + m$ and $n \times m$ rather than $+nm$ and $\times nm$ to match the usual notation in mathematics.

The axioms of **HA** are those for equality and as follows.

1. $\neg\, Sx = 0$
2. $Sx = Sy \;\to\; x = y$
3. $x + 0 \;=\; 0$
4. $x + (Sy) \;=\; S(x + y)$
5. $x \times 0 \;=\; x$
6. $x \times (Sy) \;=\; (x \times y) + x$
7. $(\varphi[x/0] \;\wedge\; \forall x\, (\varphi \to \varphi[x/S(x)])) \;\to\; \forall x\, \varphi$ for each formula φ (induction)

Although first order Heyting arithmetic is a solid base to work from, it is limited in the sense that we can only directly talk about numbers. However, many of the axioms that we will consider are most naturally stated in terms of functions or sets. We will therefore define two "higher order" extensions of **HA** where we can reason about functions or sets in addition to numbers.

2.3 Second order Heyting arithmetic

The first extension we will consider is *second order* Heyting arithmetic, which we abbreviate to **HAS**. The signature of **HAS** has two sorts: one for numbers, $\mathfrak{N}$ and one for sets of numbers, $\mathfrak{S}$. We write variables of sort $\mathfrak{N}$ using lower case letters, and variables of sort $\mathfrak{S}$ as upper case.

It has equality relations for both numbers and sets and a relation symbol $\in$. The arity of $\in$ is $\mathfrak{N}, \mathfrak{S}$. We write $\in nX$ using infix notation (i.e. as $n \in X$) to match the usual notation for set membership. Finally, **HAS** has the same operator symbols $0, S, +, \times$ as for **HA**.

The axioms of **HAS** include those of **HA** together with three more: *comprehension*, *extensionality* and *second order induction*.

Comprehension is an axiom scheme, which asserts for each formula φ the following:

$$\exists X \, \forall n \; n \in X \leftrightarrow \varphi$$

Extensionality is the following axiom:

$$\forall X, Y \, (\forall n \, n \in X \leftrightarrow n \in Y) \to (X = Y)$$

Finally, second order induction is the axiom below.

$$\forall X \, (0 \in X \; \wedge \; \forall n \, (n \in X \to Sn \in X)) \; \to \; \forall n \, n \in X$$

Note that second order induction and comprehension together imply the first order induction scheme, so we can drop first order induction from the definition without changing the set of theorems.

The richer language of **HAS** allows us to naturally state axioms that would be clumsy or even impossible to state in **HA**. For example, we can see our first example of *anti-classical* axioms, i.e. axioms that are provably false in classical logic. These are the *uniformity principle* (**UP**) and *unzerlegbarkeit* (**UZ**).

UP	$\forall X \, \exists n \, \varphi \; \to \; \exists n \, \forall X \, \varphi$
UZ	$\forall X \, (\varphi \vee \neg\varphi) \; \to \; (\forall X \, \varphi \; \vee \; \forall X \, \neg\varphi)$

We can see that **UP** implies **UZ**, and that **UZ** contradicts the law of excluded middle (for example by taking φ to be $\forall n \, n \notin X$). However, we will see later in the course that **UP** holds in *realizability* models of **HAS**, and so the theory is consistent.

2.4 Heyting arithmetic with finite types

The first extension of **HA** we are going to consider is Heyting arithmetic with *finite types*, which we abbreviate to $\mathbf{HA}_\omega$. Finite types allow us to also reason about functions. This includes functions that take a number as input and return a number as output. However, it goes much further - we can also have functions that take functions as input, and functions that take those as input, and so forth. We formalise this idea by first defining the set of *finite type symbols* as follows.

Definition 2.1. The set of *finite type symbols* is inductively generated by the following conditions:

1. N is a finite type symbol (the "type of numbers")

2. If σ and τ are type symbols, then so is $\sigma \times \tau$ (the "product type" of σ and τ)

3. If σ and τ are type symbols, then so is $\sigma \to \tau$ (the "type of functions from σ to τ")

We take the set of sorts of $\mathbf{HA}_\omega$ to be the finite type symbols.

Since we are thinking of $\sigma \to \tau$ as the sort of functions from σ to τ we should have some way to take an element of $\sigma \to \tau$ and an element of σ and return an element of τ by "function application." We do this by simply adding for all finite types σ and τ an operator symbol $\mathrm{Ap}^{\sigma,\tau}$ of arity $(\sigma \to \tau), \sigma \to \tau$. We will usually omit Ap when we write down terms of $\mathbf{HA}_\omega$. That is, for terms t, s of sorts $\sigma \to \tau$ and σ we will write $\mathrm{Ap}^{\sigma,\tau}\, ts$ simply as ts. When we have a series of function applications in a row, we follow a further notational convention of dropping parentheses according to "left associativity." Namely, for three terms r, s, t, we write rst to mean $(rs)t$, which is in turn notational shorthand for $\mathrm{Ap}(\mathrm{Ap}\, rs)t$.

Since we are thinking of $\sigma \times \tau$ as the product of σ and τ we should have some way to create new elements of $\sigma \times \tau$ by "pairing together an element of σ and an element of τ" and some way to take an element of $\sigma \times \tau$ and "project out" the component in σ and the component in τ. Again this is achieved by simply adding operator symbols. However, now we have Ap we can add them as constant symbols (i.e. 0-ary operator symbols) of the appropriate function type. Namely, we add a constant $\mathbf{p}^{\sigma,\tau}$ of sort $\sigma \to (\tau \to (\sigma \times \tau))$, a constant $\mathbf{p}_0^{\sigma,\tau}$ of sort $\sigma \times \tau \to \sigma$, and a constant $\mathbf{p}_1^{\sigma,\tau}$ of sort $\sigma \times \tau \to \tau$. In order for these constants to behave as described we will need to add the axioms below:

$$\begin{aligned} \mathbf{p}_0(\mathbf{p}xy) &= x \\ \mathbf{p}_1(\mathbf{p}xy) &= y \\ \mathbf{p}(\mathbf{p}_0 z)(\mathbf{p}_1 z) &= z \end{aligned}$$

We also add some constant symbols for generating new elements of a function sort. Firstly given an element of σ, we should get functions "constantly equal to that element." We implement this with a constant symbol $\mathbf{k}^{\sigma,\tau}$ of sort $\sigma \to (\tau \to \sigma)$. For this to behave as expected, we add the axiom below.

$$\mathbf{k}xy = x$$

We also want a way to compose two functions. It turns out that it's useful to have something a bit stronger that we can think of as "composition with a parameter." For this we add a constant symbol $\mathbf{s}^{\rho,\sigma,\tau}$ of sort $(\rho \to (\sigma \to \tau)) \to ((\rho \to \sigma) \to (\rho \to \tau))$, satisfying the axiom below:

$$\mathbf{s}xyz = xz(yz)$$

Finally, we have one more constant $\mathbf{r}^\sigma$, the *recursor*. One way to think about this is that we might want a way to "iterate" a function n times given a number

n. However, like with $\mathbf{s}$ it turns out that it's more useful to have a stronger version that takes a parameter as input. The sort of $\mathbf{r}^\sigma$ is $\sigma \to ((\sigma \to (N \to \sigma)) \to (N \to \sigma))$ and it satisfies the axioms below.

$$\mathbf{r}xy0 = x$$
$$\mathbf{r}xy(Sz) = y(\mathbf{r}xyz)z$$

We also add an equality relation for each sort, and the usual equality axioms. It also has the usual induction axiom scheme from **HA**.

Putting this all together we can formally define $\mathbf{HA}_\omega$ as follows.

Definition 2.2. *Heyting arithmetic with finite types*, $\mathbf{HA}_\omega$ is the theory defined as follows.

The set of sorts is the set of finite type symbols.

The set of operator symbols is an operator symbol $\mathrm{Ap}^{\sigma,\tau}$ of arity $(\sigma \to \tau), \sigma \to \tau$, for all finite types σ and τ together with the following constant symbols for all types σ, τ, ρ.

Constant symbol	Sort
0	N
S	$N \to N$
$\mathbf{p}^{\sigma,\tau}$	$\sigma \to (\tau \to (\sigma \times \tau))$
$\mathbf{p}_0^{\sigma,\tau}$	$\sigma \times \tau \to \sigma$
$\mathbf{p}_1^{\sigma,\tau}$	$\sigma \times \tau \to \tau$
$\mathbf{k}^{\sigma,\tau}$	$\sigma \to (\tau \to \sigma)$
$\mathbf{s}^{\rho,\sigma,\tau}$	$(\rho \to (\sigma \to \tau)) \to ((\rho \to \sigma) \to (\rho \to \tau))$
$\mathbf{r}^\sigma$	$\sigma \to ((\sigma \to (N \to \sigma)) \to (N \to \sigma))$

The only relation symbols are equality relations for each sort.

Firstly, we have the axioms for equality. In the language of $\mathbf{HA}_\omega$, this amounts to the following axioms.

$$x = x \qquad x = y \to y = x \qquad x = y \wedge y = z \to x = z$$
$$y = z \to \mathrm{Ap}\, xy = \mathrm{Ap}\, xz \qquad x = y \to \mathrm{Ap}\, xz = \mathrm{Ap}\, yz$$

Next, we have the formulas below, where we are following the conventions of writing $\mathrm{Ap}\, ts$ as ts for terms t and s, and taking application to be left associative.

$$\mathbf{p}_0(\mathbf{p}xy) = x \qquad \mathbf{p}_1(\mathbf{p}xy) = y \qquad \mathbf{p}(\mathbf{p}_0 z)(\mathbf{p}_1 z) = z$$
$$\mathbf{k}xy = x \qquad \mathbf{s}xyz = xz(yz)$$
$$\mathbf{r}xy0 = x \qquad \mathbf{r}xy(Sz) = y(\mathbf{r}xyz)z$$
$$Sx = Sy \to x = y \qquad \neg 0 = Sx$$

Finally, $\mathbf{HA}_\omega$ has the usual induction scheme from **HA**. That is, for each formula φ in the language of $\mathbf{HA}_\omega$, we have the following axiom.

$$(\varphi[x/0] \wedge \forall x\, (\varphi \to \varphi[x/S(x)])) \to \forall x\, \varphi$$

Although $\mathbf{HA}_\omega$ seems like a quite elaborate system, it is in some ways easier to deal with than **HAS**. We will see for example, that realizability models of $\mathbf{HA}_\omega$ are in a certain sense better behaved than those of of **HAS**. However, for this course, the main advantage of $\mathbf{HA}_\omega$ over **HAS** is that many of the axioms we are going to consider are stated in terms of functions. Although it is possible to implement functions as sets, using their graphs, it is more natural to just use a system that has a good notion of function already built in.

For example, we can now formulate the axiom of choice, a well known axiom usually viewed as uncontroversial today, but with an important place in the history and philosophy of mathematics. In $\mathbf{HA}_\omega$ it is most natural to not view it as a single axiom, but instead for all finite types σ and τ we have a separate axiom scheme $\mathbf{AC}^{\sigma,\tau}$ defined as the following, for each formula φ.

$$\forall x^\sigma \, \exists y^\tau \, \varphi \quad \to \quad \exists f^{\sigma\to\tau} \, \forall x^\sigma \, \varphi[y/fx]$$

We are going to see, for example,

1. using topological models, even the weakest version $\mathbf{AC}^{N,N}$ is independent of $\mathbf{HA}_\omega$
2. $\mathbf{AC}^{N,N}$ and $\mathbf{AC}^{N\to N,N}$ hold in certain realizability models and so are consistent with some anti-classical axioms.

We will also consider another axiom scheme, *function extensionality*, which is the following statement for all finite types σ and τ.

$$\forall f^{\sigma\to\tau} \, \forall g^{\sigma\to\tau} \, (\forall x^\sigma \, fx =^\tau gx) \to f =^{\sigma\to\tau} g$$

This is less often seen outside logic, since it usually viewed as obvious, and can be proved in set theory, when using the standard implementation of functions as graphs. However, it is often considered by philosophers of mathematics since it concerns the question of how we know when two objects (in this case functions) are equal to each other. One of the ideas we will see in this course is that although the axiom of choice is often seen as a non constructive axiom, this is in a certain sense only true when it is combined with function extensionality.

2.5 λ-terms in $\mathbf{HA}_\omega$

It might seem at first that $\mathbf{HA}_\omega$ is quite limited in what functions you can construct using the axioms. However, $\mathbf{s}$ and $\mathbf{k}$ turn out to be very powerful.

We first note that we can derive an "identity" term:

Definition 2.3. For each finite type, σ, we write $\mathbf{i}^\sigma$ for the closed term of $\mathbf{HA}_\omega$ defined as $\mathbf{s}^{\sigma,(\sigma\to\sigma),\sigma}\mathbf{k}^{\sigma,(\sigma\to\sigma)}\mathbf{k}^{\sigma,\sigma}$.

Proposition 2.4. *In* $\mathbf{HA}_\omega$ *we can prove* $\mathbf{i}x = x$.

Proof. By applying the axiom for $\mathbf{s}$ followed by the axiom for $\mathbf{k}$ we get the following equations:

$$\begin{aligned}\mathbf{i}x &:= \mathbf{skk}x \\ &= \mathbf{k}x(\mathbf{k}x) \\ &= x\end{aligned}$$

□

Using $\mathbf{s}$, $\mathbf{k}$ and $\mathbf{i}$ we can then show the powerful λ-abstraction lemma, that can be used to construct any function that can be written down as a term:

Lemma 2.5. *Let t be a term of $\mathbf{HA}_\omega$ of sort σ with a free variable x of sort τ. Then there is a term $\lambda x.t$ of sort $\tau \to \sigma$ satisfying the equation*

$$(\lambda x.t)y = t[x/y]$$

Proof. We construct $(\lambda x.t)$ by induction on the definition of terms.

If t is the free variable x, we take $(\lambda x.t)$ to be $\mathbf{i}^\sigma$, and it is clear this satisfies the lemma. (Note that in this case we have $\sigma = \tau$.)

If t is a constant c or free variable other than x, say y, (necessarily of sort σ) then we take $(\lambda x.t)$ to be $\mathbf{k}^{\tau,\sigma}c$ or $\mathbf{k}^{\tau,\sigma}y$ respectively. We again note that it is clear this satisfies the lemma, by the axiom for $\mathbf{k}$.

Finally, if t is of the form rs for a term r of sort $\rho \to \sigma$ and a term s of sort ρ, then we take $\lambda x.t$ to be $\mathbf{s}^{\tau,\rho,\sigma}(\lambda x.r)(\lambda x.s)$. We check that this works as follows.

$$\begin{aligned}(\lambda x.t)y &:= \mathbf{s}(\lambda x.r)(\lambda x.s)y & \\ &= ((\lambda x.r)y)((\lambda x.s)y) && \text{axiom for } \mathbf{s} \\ &= r[x/y]\, s[x/y] && \text{inductive hypothesis} \\ &= (rs)[x/y] &&\end{aligned}$$

□