Peano Arithmetic

Computability and Logic

Peano Arithmetic

- Language of Arithmetic:
  - One constants: **0**
  - Three function symbols: **s**, **+**, and **×**
- 6 Axioms:
  - **PA1:** $\forall x \quad s(x) \neq 0$
  - **PA2:** $\forall x \, \forall y \quad (s(x) = s(y) \rightarrow x = y)$
  - **PA3:** $\forall x \quad x + 0 = x$
  - **PA4:** $\forall x \, \forall y \quad x + s(y) = s(x + y)$
  - **PA5:** $\forall x \quad x \times 0 = 0$
  - **PA6:** $\forall x \, \forall y \quad x \times s(y) = (x \times y) + x$

Standard Interpretation N

- N is the following (standard) interpretation of the language of Arithmetic:
  - Domain: natural numbers (0,1,2,3, etc)
  - N(**0**) = 0
  - N(**s**) = s, the successor function
  - N(**+**) = +, the addition function
  - N(**×**) = ×, the multiplication function
  - More technically, where $\mathbf{t_0}$, $\mathbf{t_1}$, and $\mathbf{t_2}$ are variable-free terms:
    - N(**s(t$_0$)**) = s(N(**t$_0$**))
    - N($\mathbf{t_1 + t_2}$) = N($\mathbf{t_1}$) + N($\mathbf{t_2}$)
    - N($\mathbf{t_1 \times t_2}$) = N($\mathbf{t_1}$) × N($\mathbf{t_2}$)

What can be Proven in PA?

- **s(s(0)) + s(s(0)) = s(s(s(s(0))))** (i.e. “2 + 2 = 4”)
- … (“(2 × 4) + 1 = 3 × 3”)
- In fact: Where **n** is the expression **s(s(…s(0)..))** (n successor functions), you can prove in PA:
  - $\underline{\mathbf{n}}_1$ **+** $\underline{\mathbf{n}}_2$ **=** $\underline{\mathbf{n}}$ for any n_1 and n_2 where $n_1 + n_2 = n$
  - $\underline{\mathbf{n}}_1$ **×** $\underline{\mathbf{n}}_2$ **=** $\underline{\mathbf{n}}$ for any n_1 and n_2 where $n_1 \times n_2 = n$
  - **t =** $\underline{\mathbf{n}}$ for any expression **t** for which N(**t**) = n
  - Hence: $\mathbf{t}_1 = \mathbf{t}_2$ for any expressions $\mathbf{t}_1$ and $\mathbf{t}_2$ for which N($\mathbf{t}_1$) = N($\mathbf{t}_2$). Therefore, also $\mathbf{t}_1 + \mathbf{t}_2 = \mathbf{t}_2 + \mathbf{t}_1$, etc.
- In short: any arithmetical truth that can be expressed in LA without the use of quantifiers!

‘Proven in PA’

- By the way, we have to be careful with our language here:
  - When we say ‘P can be proven in PA’, we mean ‘P can be derived from the axioms of PA’ … but that all depends on what particular proof system you use.
  - Assuming we use proof system F, we thus really mean: ‘P can be derived, in system F, from the axioms of PA’

Two Very Important Properties

- For every deductive system of formal logic *S* we can define the following 2 properties:
  - 1. *Soundness*: A system *S* is sound iff for any Γ and ψ:
    - if $\Gamma \vdash_S \psi$ then $\Gamma \vDash \psi$
  - 2. *Completeness*: A system *S* is complete iff for any Γ and ψ :
    - if $\Gamma \vDash_{TF} \psi$ then $\Gamma \vdash_S \psi$
- As it turns out, proof system F is both sound and complete (take Intermediate Logic for proofs of these results)

What Cannot Be Proven in PA?

- Many things that are true in arithmetic *cannot* be proven in PA.
- Example:
  - $\forall \mathbf{x}\ \forall \mathbf{y}\ \mathbf{x + y = y + x}$
  - See handout ‘Non-Standard Models’
    - The handout shows that *not* PA $\vDash \forall \mathbf{x}\ \forall \mathbf{y}\ \mathbf{x + y = y + x}$
    - Since F is sound, that means *not* PA $\vdash_F \forall \mathbf{x}\ \forall \mathbf{y}\ \mathbf{x + y = y + x}$
- This is interesting since, as we just saw, we *can* prove for any two terms $\mathbf{t_1}$ and $\mathbf{t_2}$: $\mathbf{t_1 + t_2 = t_2 + t_1}$
- Make sure you understand the difference!

Mathematical Induction

- Fortunately, we can prove many more things by adding the following axiom of induction:
  - $(\varphi(0) \wedge \forall x\, (\varphi(x) \rightarrow \varphi(s(x)))) \rightarrow \forall x\, \varphi(x)$

- Note: this is really an axiom scheme, representing an infinite number of axioms, since there are an infinite number of formulas $\varphi(x)$ that have x as their only free variable.

Strong Induction

- Sometimes, it is helpful to have strong induction, which can be formalized as:
  - $\forall x\ (\forall y\ (y < x \rightarrow \varphi(y)) \rightarrow \varphi(x)) \rightarrow \forall x\ \varphi(x)$

- Note that we use a '< ' here, but this is not part of our original language. We can do 2 things:
  1. Consider 'y < x' to be short-hand for an expression that *is* part of our original language, e.g. we could say that 'x < y' is shorthand for $\exists z\ x + s(z) = y$
  2. Add '<' as a 2-place predicate to our language, and add a definitional axiom to our axioms, such as $\forall x\ \forall y\ (x < y \leftrightarrow \exists z\ x + s(z) = y)$

Let’s do Some Proofs!

- $\forall x\ 0 + x = x$ (16.29)
- $\forall x\ x + s(0) = s(x)$
- $\forall x\ s(0) + x = s(x)$
- $\forall x\ x \times s(0) = x$
- $\forall x\ s(0) \times x = x$ (16.30)
- $\forall x\ \forall y\ s(x) + y = s(x + y)$ (addition left recursion)
- $\forall x\ \forall y\ x + y = y + x$ (16.36)
- $\forall x\ \forall y\ s(x) \times y = (x \times y) + y$ (multiplication left recursion; 16.37)
- $\forall x\ \forall y\ x \times y = y \times x$ (16.38)

Some More

- $\forall x\ s(x) \neq x$ (Successor Lemma)
- $\forall x\ (x \neq 0 \rightarrow \exists y\ s(y) = x)$ (Predecessor Lemma; also see 16.50 and 16.43)
- $\forall x\ x < s(x)$ (16.40)
- $\forall x\ s(x) < s(y) \rightarrow x < y$ (other way around from 16.44 … is this the one the book meant?)
- $\forall x \neg x < x$ (16.45)