FLOLAC'14

Type theory and logic

Lecture III: natural number arithmetic

3 July 2014

柯向上

Department of Computer Science
University of Oxford

Hsiang-Shang.Ko@cs.ox.ac.uk

Natural numbers

- Formation:

$$\frac{}{\Gamma \ \vdash \ \mathbb{N} : \mathcal{U}} \ (\mathbb{N}\mathsf{F})$$

- Introduction:

$$\frac{}{\Gamma \ \vdash \ \texttt{zero} : \mathbb{N}} \ (\mathbb{N}\mathsf{IZ}) \qquad \frac{\Gamma \ \vdash \ n : \mathbb{N}}{\Gamma \ \vdash \ \texttt{suc}\ n : \mathbb{N}} \ (\mathbb{N}\mathsf{IS})$$

- Elimination:

$$\frac{\begin{array}{l} \Gamma \ \vdash \ P : \mathbb{N} \to \mathcal{U} \\ \Gamma \ \vdash \ z : P\ \texttt{zero} \\ \Gamma \ \vdash \ s : \Pi[x : \mathbb{N}]\ P\,x \to P\,(\texttt{suc}\ x) \\ \Gamma \ \vdash \ n : \mathbb{N} \end{array}}{\Gamma \ \vdash \ \texttt{ind}\ P\,z\,s\,n : P\ n} \ (\mathbb{N}\mathsf{E})$$

Logically this is the *induction principle*; computationally this is *primitive recursion*.

Natural numbers — computation rules

- Computation:

$$\frac{\begin{array}{l}\Gamma \ \vdash \ P : \mathbb{N} \to \mathcal{U} \\ \Gamma \ \vdash \ z : P\ \texttt{zero} \\ \Gamma \ \vdash \ s : \Pi[x : \mathbb{N}]\ P\,x \to P\,(\texttt{suc}\ x)\end{array}}{\Gamma \ \vdash \ \texttt{ind}\ P\,z\,s\ \texttt{zero} = z \ \in \ P\ \texttt{zero}}\ (\mathbb{N}\mathsf{CZ})$$

$$\frac{\begin{array}{l}\Gamma \ \vdash \ P : \mathbb{N} \to \mathcal{U} \\ \Gamma \ \vdash \ z : P\ \texttt{zero} \\ \Gamma \ \vdash \ s : \Pi[x : \mathbb{N}]\ P\,x \to P\,(\texttt{suc}\ x) \\ \Gamma \ \vdash \ n : \mathbb{N}\end{array}}{\Gamma \ \vdash \ \texttt{ind}\ P\,z\,s\,(\texttt{suc}\ n) = s\,n\,(\texttt{ind}\ P\,z\,s\,n) \ \in \ P\,(\texttt{suc}\ n)}\ (\mathbb{N}\mathsf{CS})$$

Exercise. Define addition and multiplication with `ind`.

Induction principle

The set of natural numbers is *inductively defined*.

Identity types

Identity types are also called *propositional equality*, especially when drawing contrast with judgemental equality.

- Formation:

$$\frac{\Gamma \vdash A : \mathcal{U} \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : A}{\Gamma \vdash \mathtt{Id}\ A\ t\ u : \mathcal{U}}\ (\mathtt{IdF})$$

- Introduction:

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \mathtt{refl}\ t : \mathtt{Id}\ A\ t\ t}\ (\mathtt{IdI})$$

Exercise. Assume $\Gamma \vdash t = u \in A$ and derive $\Gamma \vdash \mathtt{refl}\ t : \mathtt{Id}\ A\ t\ u$.

Identity types — elimination and computation

- Elimination:

$$\frac{\begin{array}{l}\Gamma \;\vdash\; P : A \to \mathcal{U} \\ \Gamma \;\vdash\; t : A \\ \Gamma \;\vdash\; p : P\,t \\ \Gamma \;\vdash\; u : A \\ \Gamma \;\vdash\; q : \mathtt{Id}\;A\;t\;u\end{array}}{\Gamma \;\vdash\; \mathtt{transport}\;P\;p\;q \;:\; P\;u}\;(\mathtt{IdE})$$

- Computation:

$$\frac{\Gamma \;\vdash\; P : A \to \mathcal{U} \qquad \Gamma \;\vdash\; t : A \qquad \Gamma \;\vdash\; p : P\,t}{\Gamma \;\vdash\; \mathtt{transport}\;P\;p\;(\mathtt{refl}\;t) = p \;\in\; P\;t}\;(\mathtt{IdC})$$

$\texttt{Id}$ is an equivalence relation

$\texttt{Id}$ is obviously reflexive — we can derive

$$\vdash \lambda A.\, \lambda x.\, \texttt{refl}\, x : \Pi[A : \mathcal{U}]\ \Pi[x : A]\ \texttt{Id}\, A\, x\, x$$

Exercise. Prove that $\texttt{Id}$ is symmetric and transitive, i.e.,

$$\Pi[A : \mathcal{U}]\ \Pi[x : A]\ \Pi[y : A]\ \texttt{Id}\, A\, x\, y \to \texttt{Id}\, A\, y\, x$$

and

$$\begin{gathered}\Pi[A : \mathcal{U}]\ \Pi[x : A]\ \Pi[y : A]\ \Pi[z : A] \\ \texttt{Id}\, A\, x\, y \to \texttt{Id}\, A\, y\, z \to \texttt{Id}\, A\, x\, z\end{gathered}$$

Identity types — general elimination and computation

- Elimination:

$$\frac{\begin{array}{lll}\Gamma & \vdash & t : A \\ \Gamma & \vdash & P : \Pi[x : A]\ \mathtt{Id}\ A\ t\ x \to \mathcal{U} \\ \Gamma & \vdash & p : P\ t\ (\mathtt{refl}\ t) \\ \Gamma & \vdash & u : A \\ \Gamma & \vdash & q : \mathtt{Id}\ A\ t\ u\end{array}}{\Gamma \ \vdash\ \mathtt{J}\ P\ p\ q : P\ u\ q}\ (\mathtt{IdE})$$

- Computation:

$$\frac{\begin{array}{lll}\Gamma & \vdash & t : A \\ \Gamma & \vdash & P : \Pi[x : A]\ \mathtt{Id}\ A\ t\ x \to \mathcal{U} \\ \Gamma & \vdash & p : P\ t\ (\mathtt{refl}\ t)\end{array}}{\Gamma \ \vdash\ \mathtt{J}\ P\ p\ (\mathtt{refl}\ t) = p\ \in P\ t\ (\mathtt{refl}\ t)}\ (\mathtt{IdC})$$

Peano axioms

Peano axioms specify an *equational theory* of natural number arithmetic; all of them are provable in type theory.

- Zero is a natural number. If n is a natural number, so is the successor of n.
  — The introduction rules.
- Equality on natural numbers is an equivalence relation; that is, it is reflexive, transitive, and symmetric.
  — We use `Id`, which indeed satisfies the above properties.
- The successor operation is an injective function, i.e.,

$$\Pi\,[\,m : \mathbb{N}\,]\ \Pi\,[\,n : \mathbb{N}\,]\ \mathtt{Id}\ \mathbb{N}\ m\ n \leftrightarrow \mathtt{Id}\ \mathbb{N}\ (\mathtt{suc}\ m)\ (\mathtt{suc}\ n)$$

- The successor operation never yields zero, i.e.,

$$\Pi\,[\,n : \mathbb{N}\,]\ \mathtt{Id}\ \mathbb{N}\ (\mathtt{suc}\ n)\ \mathtt{zero} \to \bot$$

Peano axioms

- Addition satisfies

$$\Pi\,[\,n : \mathbb{N}\,]\ \mathtt{Id}\ \mathbb{N}\ (\mathtt{zero} + n)\ n$$

  and

$$\Pi\,[\,m : \mathbb{N}\,]\ \Pi\,[\,n : \mathbb{N}\,]\ \mathtt{Id}\ \mathbb{N}\ ((\mathtt{suc}\ m) + n)\ (\mathtt{suc}\ (m + n))$$

- Multiplication satisfies

$$\Pi\,[\,n : \mathbb{N}\,]\ \mathtt{Id}\ \mathbb{N}\ (\mathtt{zero} \times n)\ \mathtt{zero}$$

  and

$$\Pi\,[\,m : \mathbb{N}\,]\ \Pi\,[\,n : \mathbb{N}\,]\ \mathtt{Id}\ \mathbb{N}\ ((\mathtt{suc}\ m) \times n)\ (n + m \times n)$$

- The induction principle holds for natural numbers.
  — The elimination rule.

Computational foundation

In type theory, Peano “axioms” are merely consequences, and do not really play an important role.

We now have a more natural foundation based on the idea of typed computation.

The infamous proof of $1 + 1 = 2$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\mathbf{HA} \vdash \textit{transitivity}}}{\mathbf{HA} \vdash \forall \mathtt{y}.\ \forall \mathtt{z}.\ \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{y} \wedge \mathtt{y} \equiv \mathtt{z} \to \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{z}}\,(\forall\mathrm{E})}{\mathbf{HA} \vdash \forall \mathtt{z}.\ \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \wedge \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \equiv \mathtt{z} \to \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{z}}\,(\forall\mathrm{E})}{\mathbf{HA} \vdash \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \wedge \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \equiv \mathtt{suc\ (suc\ zero)} \to \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{suc\ (suc\ zero)}}\,(\forall\mathrm{E}) \quad \dfrac{\dfrac{\dfrac{\dfrac{}{\mathbf{HA} \vdash \textit{additionS}}}{\mathbf{HA} \vdash \forall \mathtt{y}.\ \mathtt{suc\ zero} + \mathtt{y} \equiv \mathtt{suc\ (zero} + \mathtt{y)}}\,(\forall\mathrm{E})}{\mathbf{HA} \vdash \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{suc\ (zero} + \mathtt{suc\ zero)}}\,(\forall\mathrm{E}) \quad \dfrac{\dfrac{\dfrac{\dfrac{}{\mathbf{HA} \vdash \textit{congruence}}}{\mathbf{HA} \vdash \forall \mathtt{y}.\ \mathtt{zero} + \mathtt{suc\ zero} \equiv \mathtt{y} \to \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \equiv \mathtt{suc\ y}}\,(\forall\mathrm{E})}{\mathbf{HA} \vdash \mathtt{zero} + \mathtt{suc\ zero} \equiv \mathtt{suc\ zero} \to \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \equiv \mathtt{suc\ (suc\ zero)}}\,(\forall\mathrm{E}) \quad \dfrac{\dfrac{}{\mathbf{HA} \vdash \textit{additionZ}}}{\mathbf{HA} \vdash \mathtt{zero} + \mathtt{suc\ zero} \equiv \mathtt{suc\ zero}}\,(\forall\mathrm{E})}{\mathbf{HA} \vdash \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \equiv \mathtt{suc\ (suc\ zero)}}\,(\to\mathrm{E})}{\mathbf{HA} \vdash \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \wedge \mathtt{suc\ (zero} + \mathtt{suc\ zero)} \equiv \mathtt{suc\ (suc\ zero)}}\,(\wedge\mathrm{I})}{\mathbf{HA} \vdash \mathtt{suc\ zero} + \mathtt{suc\ zero} \equiv \mathtt{suc\ (suc\ zero)}}\,(\to\mathrm{E})$$

is entirely reduced to computation in type theory.