Introduction to Type-Level Programming

Alexander Lehmann <lehmanna at in.tum.de>

February 18th, 2012

Type-Level Programming

- Evaluation at compile-time
- Maximum type safety
- Scala's type system turing-complete (*michid*)
  http://stackoverflow.com/a/4050833/271678
- "Object oriented" vs. "functional style" (*dsg*)
  http://stackoverflow.com/a/4443972/271678

- Implementation based on "shapeless" by *Miles Sabin*

  https://github.com/milessabin/shapeless

- Suggested reading (*Mark Harrah*)

  http://apocalisp.wordpress.com/2010/06/08/type-level-programming-in-scala/

Objectives

- Type-level representation of natural numbers
- Value representation of type-level entities
- Basic arithmetics (add, mult, factorial)
- Why the !@#^ is everybody talking about Peano?

Peano Axiomes

- Formalization of the set of natural numbers

(1) $0 \in \mathbb{N}$

(2) $n \in \mathbb{N} \Rightarrow n' \in \mathbb{N}$

(3) $n \in \mathbb{N} \Rightarrow n' \neq 0$

(4) $m, n \in \mathbb{N} \Rightarrow (m' = n' \Rightarrow m = n)$

(5)

Peano Axiomes

- Formalization of the set of natural numbers

(1) $0 \in \mathbb{N}$

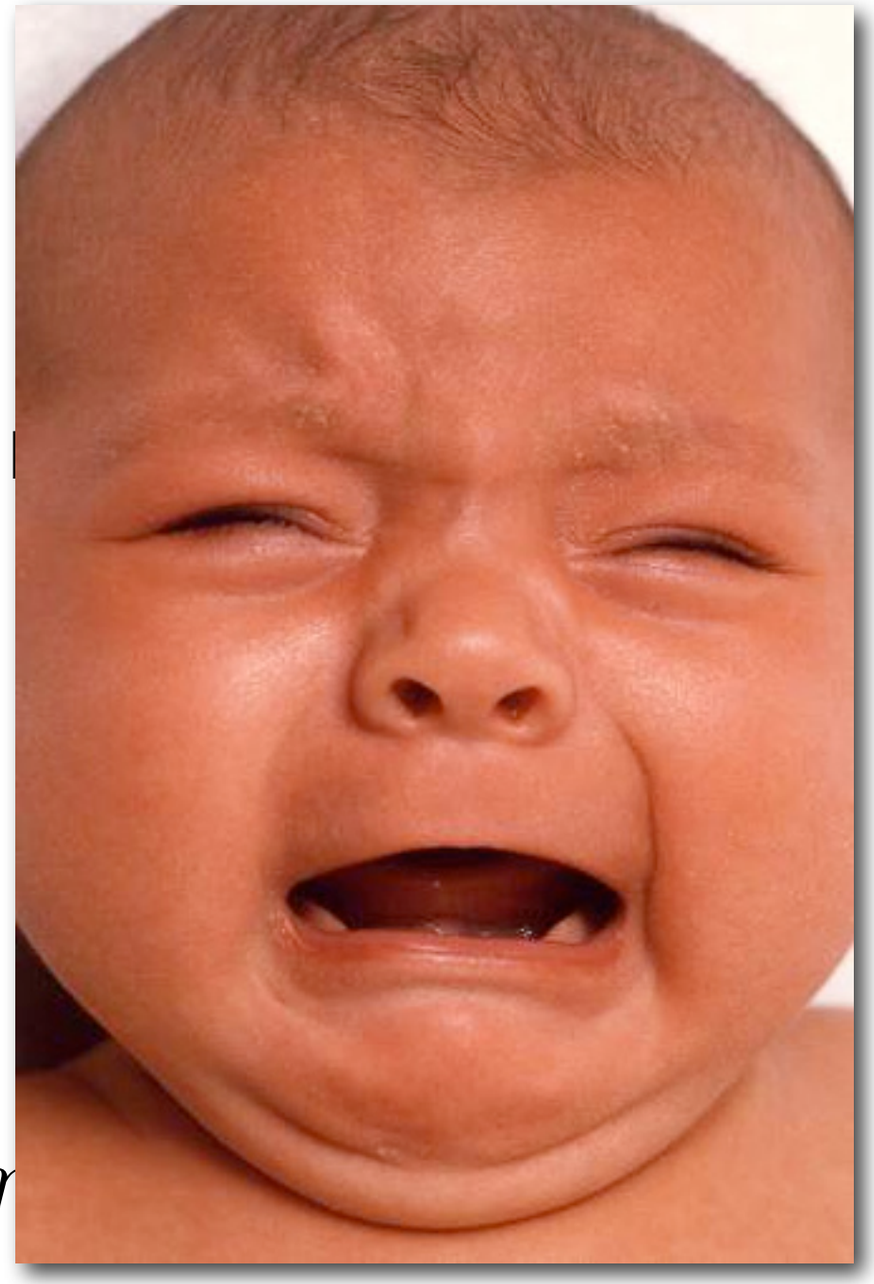
(2) $n \in \mathbb{N} \Rightarrow n' \in \mathbb{N}$

(3) $n \in \mathbb{N} \Rightarrow n' \neq 0$

(4) $m, n \in \mathbb{N} \Rightarrow (m' = n' \Rightarrow m = n)$

(5) $0 \in X \wedge \forall n \in \mathbb{N} : (n \in X \Rightarrow n' \in X) \Rightarrow \mathbb{N} \subseteq X$

Peano Axio



- Formalization of the set of

(1) $0 \in \mathbb{N}$

(2) $n \in \mathbb{N} \Rightarrow n' \in \mathbb{N}$

(3) $n \in \mathbb{N} \Rightarrow n' \neq 0$

(4) $m, n \in \mathbb{N} \Rightarrow (m' = n' \Rightarrow n$

(5) $0 \in X \wedge \forall n \in \mathbb{N} : (n \in X \Rightarrow n' \in X) \Rightarrow \mathbb{N} \subseteq X$

Peano Axiomes

If some predicate S

- is true for 0 and
- if from the fact that S is true for n it follows that it's also true for n+1

then it's also true for every natural number.

→ "Principle of induction"

Types to Values

Using the "induction axiom", we get

Base case:

$n = 0 \quad \Rightarrow \quad 0$

Induction step:

$n \rightarrow n+1 \quad \Rightarrow \quad v(n+1) = v(n) + 1$

Summation

- Recursive definition

$$\begin{aligned} n + 0 &:= n \\ n + m' &:= (n + m)' \end{aligned}$$

- Example

Summation

- Recursive definition

$$\begin{aligned} n + 0 &:= n \\ n + m' &:= (n + m)' \end{aligned}$$

- Example

$$a + 1 = a + 0' = (\underbrace{a + 0}_{=:a})' = a'$$

Multiplication

- Recursive definition

$$
\begin{aligned}
n \cdot 0 &:= 0 \\
n \cdot m' &:= n \cdot m + n
\end{aligned}
$$

Multiplication

- Example

$$
\begin{aligned}
a \cdot 2 &= a \cdot 1' \\
&= a \cdot 1 + a \\
&= a \cdot 0' + a \\
&= \underbrace{a \cdot 0}_{=:0} + a + a \\
&= a + a
\end{aligned}
$$

Factorial

- Recursive definition

$$
\begin{aligned}
0! &:= 1 \\
n'! &:= n! \cdot n'
\end{aligned}
$$

Factorial

- Example

$$
\begin{aligned}
3! &= 2'! \\
&= 2! \cdot 2' \\
&= 1'! \cdot 2' \\
&= 1! \cdot 1' \cdot 2' \\
&= 0'! \cdot 1' \cdot' 2 \\
&= \underbrace{0!}_{=:1} \cdot \underbrace{0'}_{=:1} \cdot \underbrace{1'}_{=:2} \cdot \underbrace{2'}_{=:3} \\
&= 6
\end{aligned}
$$

Is all this useful anyway?

- Heterogeneous Lists
  https://github.com/milessabin/shapeless/blob/master/src/main/scala/shapeless/hlist.scala
- Unboxed union types
  http://www.chuusai.com/2011/06/09/scala-union-types-curry-howard/
- Builders
  http://www.scala-lang.org/docu/files/collections-api/collections-impl.html