Type Theory in Rosario

Thorsten Altenkirch

July 25, 2011

Zermelo-Fraenkel Set Theory

Zermelo (1871-1953) Fraenkel (1891-1965)

- Axiomatic Set Theory $\approx$ 1925
- ZFC = Zermelo-Fraenkel with Axiom of Choice
- Foundations of modern Mathematics
- Additional axioms, e.g. the continuum hypothesis

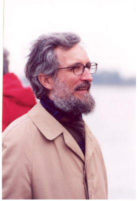
Set Theory for Computer Science?

- Set Theory is untyped (everything is a set), while programming languages are typed (either statically or dynamically).
- Basic concepts from computer science (records, functions) are not primitive in Set Theory.
- Basic operations in set theory (e.g. $\cap$, $\cup$) are not directly available on types.
- Set Theory is not constructive, i.e. there is a set theoretic *function* solving the Halting Problem.

Question:

Is there an alternative to Set Theory?

Martin-Löf Type Theory



Per Martin-Löf (1942-)

- Martin-Löf introduced Type Theory as a *constructive foundation of Mathematics* since 1972.
- Type Theory doesn't rely on predicate logic but uses types to represent propositions.
- Basic operations on types are Π-types (dependent function types) and Σ-types (dependent records).
- Type Theory is a programming language.

Propositions as types (The Curry-Howard Isomorphism)

- A proposition corresponds to the types of it proofs.
- A proposition is true if the corresponding type is non-empty.
- Conjunction $A \wedge B$ is repesented by cartesian product ($A \times B$).
- Implication $A \to B$ is represented by function types $A \to B$ (looks the same).
- $\forall$ and $\exists$ correspond to Π (depednent function) and Σ (dependent records).

Agda

Ulf Norell

- Ulf Norell has implemented Agda, a functional programming language based on Type Theory in his PhD in 2007.
- Agda is inspired by earlier systems such as Epigram, Cayenne and Coq.
- Agda can be used to program and to reason.

Course contents

Monday	Agda intro, Programming with dependent types
Tuesday	Propositions as types
Wednesday	Mixing programming and reasoning
Thursday	Inductive relations, proof trees
Friday	Safe evaluation and typed assembly language

Material, links, exercises:

`http://www.cs.nott.ac.uk/~txa/rosario/`