Some early ideas about types

Dave MacQueen
WG2.8 2020 (Zion)
2020-03-12

Gottlob Frege, ***Begriffsschrift***, 1879

Propositional functions:
  a proposition viewed as a function applied to arguments.

Objects:
  things that propositional functions can be applied to.

Examples:

  The sky is blue.   - a proposition, can be true or false

Viewing it as a propositional function: What is the argument?

  sky has_color (blue)   -- blue is the argument

    Φ(A) == sky has color A

  (sky) has_color blue   -- sky is the argument

    Φ(A) == A has color blue

  (sky) has_color (blue) -- both sky and blue are argument
              (2-arg prop function)

  x < 3, a propositional function of one argument, x

Propositional functions are *intentional:*

The application of a propositional function produces a proposition (*statement) which can be true or false*.

Propositional functions may have several arguments

A has_color B

Propositional functions as variables in propositions (almost).

Φ(A,B)

Here Φ represents an *indeterminate propositional function* of two arguments, i.e. a variable ranging over propositional functions, hinting at propositional functions as arguments.

Can propositional functions be "objects" that other propositional functions apply to? e.g. $\Psi(\Phi)$

No. Propositional functions take "objects" as arguments, and propositional functions are not considered ”objects".

Later, in **Foundations of Arithmetic** (*1893*), Frege allowed the “*course of values*” (i.e. extension, or graph) of a propositional function to be treated as an object, and this opened the door for Russell’s paradox: A propositional function might be applied to its own extension as an object.

Foreshadowing of types in Frege’s treatment of propositional functions:

* Multi-argument types apply to pairs, triples, etc, suggesting product types

* First-order pfs apply to “individuals”, 2nd order pfs apply to first-order pfs, etc.

Russell's paradox, discovered June, 1901

$S = \{ x \mid x \notin x \}$ $S \in S \iff S \notin S$

$\Psi(\Phi)$ iff not $\Phi(|\Phi|)$. $\Psi(|\Psi|) \iff$ not $\Psi(|\Psi|)$
where $|\Psi|$ is the extension of Ψ

* Correspondence between Russell and Frege, June 1992.

Bertrand Russell, **Principles of Mathematics**,1903, Appendix B

Here Russell first proposes types as a potential solution to the paradoxes.

Preface:

"In the case of classes, I must confess, I have failed to perceive any concept fulfilling the conditions requisite for the notion of *class*. And the contradiction discussed in Chapter x. proves that something is amiss,
but what it is I have hitherto failed to discover."

Appendix B: toward a *theory of types*

"Every propositional function $\Phi(x)$ — so it is contended — has, in addition to its range of truth, a range of significance, i.e. a range within which x must lie if $\Phi(x)$ is to be a proposition at all, whether true or false. This is the first point in the theory of types; the second point is that ranges of significance form types, i.e. if x belongs to the range of significance of $\Phi(x)$, then there is a class of objects, the type of x, all of which must also belong to the range of significance of $\Phi(x)$, however Φ may be varied; and the range of significance is always either a single type or a sum of several whole types. The second point is less precise than the first, and the case of numbers introduces difficulties; but in what follows its importance and meaning will, I hope, become plainer."

Thus a type is viewed as the ***range of significance*** of a propositional function (not the range of *truth* of a propositional function).

Products: propositional functions over two or three arguments
=> The range consists of pairs, triples, respectively.

Russell developed types further in
“Mathematical logic as based on the theory of types”, 1908.

Claims *impredicativity* is at the root of all the paradoxes
— a kind of self-referential definition.

Russell’s **Vicious Circle Principle:**

> “Whatever involves all of a collection must not be one of the collection.”

He avoids impredicativity by defining orders of propositions, where a proposition is of *order* n+1 if it contains a universal quantifier over variables ranging over things of order n

This was called *ramification (*or *ramified types*) when used in conjunction with a *simple type theory* in **Principia Mathematica** (1910-1913).

The simple type theory was not defined, but involved

* products — types consisting of tuples

* higher “kinds”: 1st order pfs, 2nd order pfs, etc.

Types are not defined, nor is there a notation to express them.
Only a of “being of the same type” is defined (incompletely).

1920s: Ramsey and Hilbert and Ackermann

* The simple theory of types suffices to avoid the paradoxes.

* Ramsey (1926) gave an explicit definition of simple types

  - 0 is a simple type (ST)
  - t1, …, tn are ST => (t1,…tn) is a ST

  These describe the argument types of propositional functions.

  E.g. (0, (0,0))
  = the type of a pf that takes an individual
  and a pf with two individual arguments

1934: Curry: *Functionality* in Combinatory Logic
Curry's type theory for Combinatory Logic

* primitive F combinator for constructing function types

  Fabf ≃ f : a → b

  Semantics: $(\forall x)(x \in a \Longrightarrow f(x) \in b)$

  If x is a type, y a term, xy asserts that y has type x, $(y \in x)$

  Axiom F: $(\forall x,y,z)(Fxyz \Longrightarrow (\forall u)(xu \Longrightarrow y(zu))$

* types used as predicates (propositional functions) that can apply to "value" expressions. Distinction between variables representing types and variables representing values is implicit.

* types are things that can be asserted (proved) to apply to terms

* combinators (hence terms) can have many types (polymorphism!)
  Typing Axioms assign *type schemes* with universally
  quantified type variables (polymorphic types) to basic combinators:

[FK] $(\forall x, y)\ Fy(Fxy)K$ K : $(\forall\ x, y)\ y \to (x \to y)$

Note: this paper also defines a precursor of the Y combinator, used to show that Russell's paradox is avoided, because of the types.

1940: Church: The Simple Theory of Types

simple theory of types combined with the lambda calculus

still treated as a logical language for reasoning

two primitive types, $\boldsymbol{\iota}$ (individuals), $\boldsymbol{o}$ (propositions, or Bool) and function types, e.g.

$= : \boldsymbol{\iota} \to \boldsymbol{\iota} \to \boldsymbol{o}$ (a binary relation on individuals)

$\neg : \boldsymbol{o} \to \boldsymbol{o}$

$\forall : (\boldsymbol{\iota} \to \boldsymbol{o}) \to \boldsymbol{o}$ (universal quantification)

1969: Curry: Modified basic *functionality* in Combinatory Logic

1. types for combinators are called *functional characters*

2. type expressions are called **F-obs**: Fab = a $\to$ b

ranged over by metavariable χ

3. primitive types are called F-simples (e.g. N = Nat)

4. "indeterminate F-simples" = parameters = type variables

5. "F-schemes" are type expressions possibly containing type variables (parameters)

6. typing judgement of form $\vdash \chi$ X, where χ is an F-ob and X is a combinator term

   χ is a "functional character of X", i.e. a type (scheme) for X

   "X is stratified" = "X is well-typed" = there exists χ s.t. |- χ X

Rules: There is a single rule, namely

RULE F:

$\vdash (F\ \chi\ \epsilon)\ X\ \ \&\ \vdash \chi\ Y \implies \vdash \epsilon\ (XY)$

This is the usual -> elimination rule for function application.

Typing rules for primitive (constant) combinators I, K, and S given by the axioms:

[FI] $\vdash F\ \alpha\ \alpha\ I$ i.e. $I : \alpha \rightarrow \alpha$

[FK] $\vdash F\ \alpha\ (F\ \beta\ \alpha)\ K$ i.e. $K : \alpha \rightarrow (\beta \rightarrow \alpha)$

[FS] $\vdash F\ (F\ \alpha\ (F\ \beta\ \gamma))\ (F\ (F\ \alpha\ \beta)\ (F\ \alpha\ \gamma))\ S$

i.e. $S : (\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow (\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \gamma$