Philosophy 240: Symbolic Logic

Russell Marcus
Hamilton College
Fall 2011

Class 40 - Functions

Final Exam

Thursday

December 15

7pm - 10pm

A Motivating Argument for Functions

1. No odd numbers are even.
2. One is odd.
3. One is the square of one.
So, not all square numbers are even.

- We can regiment into **F**.
  1. $(\forall x)(Ox \supset \sim Ex)$
  2. Oo
  3. $(\exists x)[Sxo \bullet (\forall y)(Syo \supset y=x) \bullet x=o]$

  $/ \sim(\forall x)[(Sx \bullet Nx) \supset Ex]$
- But, there is a more efficient, and more fecund, option.
- Take ‘the square of x’ as a function.

Functions

- A small extension of **F** introduces functors to represent functions.
- A functions takes one or more arguments and returns a single output, its range.
- Mathematics
  - linear functions
  - exponential functions
  - periodic functions
  - quadratic functions
  - trigonometric functions.
- Science
  - force is a function of mass and acceleration
  - momentum is a function of mass and velocity
  - genetic code
- Logic
  - semantics for **PL**
- Natural language
  - the father of
  - the teacher of
- One-place functions take one argument, two-place functions take two arguments, n-place functions take n arguments.

Some Functions and Their Logical Representations

- the father of: f(x)
- the successor of: g(x)
- the sum of: f(x,y)
- the teacher of: $g(x_1...x_n)$
- These are not functions:
  - the parents of a
  - the classes that a and b share
  - the square root of x

Vocabulary of FF

- Capital letters A...Z, used as predicates
- Lower case letters
  - a, b, c, d, e, i, j, k...u are used as constants.
  - **f, g, and h are used as functors.**
  - v, w, x, y, z are used as variables.
- Five connectives: ~, •, ∨, ⊃ ≡
- Quantifiers: ∃, ∀
- Punctuation: (), [], {}

N-Tuples

- A functor term is a functor symbol followed by an n-tuple of singular terms.
- An **n-tuple of singular terms** is an ordered series of terms.
  - Singular terms: constants, variables, or functor terms
  - ‘single’, ‘double’, ‘triple’, ‘quadruple’, etc. are n-tuples.
  - Functions can take any number of arguments.
- We n-tuples in the semantics of relational predicates.
- Often: <a, b, c>
- We will represent n-tuples merely by listing the terms separated by commas.
- Some n-tuples
  - a,b
  - a,a,f(a)
  - x,y,b,d,f(x),f(a,b,f(x))
  - a

Functor Terms

- If α is an n-tuple of singular terms, then the following are all **functor terms**:
  - f(α)
  - g(α)
  - h(α)
- Note that an n-tuple of terms can include functor terms.
- ‘Functor term’ is defined recursively, which allows for composition of functions.
- For example, one can refer to the grandfather of x, using just the functions for father, e.g. ‘f(x)’, and mother, e.g. ‘g(x)’:
  - f(f(x))
  - f(g(x))
- Composition of mathematical functions
  - Take ‘h(x)’ to represent the square of x
  - ‘h(h(h(x)))’ represents the eighth power of x, i.e. $((x^2)^2)^2$.

Formation Rules for Wffs of FF

1. An n-place predicate followed by n singular terms (constants, variables, **or functor terms**) is a wff.

2. For any variable β, if α is a wff that does not contain either ‘(∃β)’ or ‘(∀β)’, then ‘(∃β)α’ and‘(∀β)α’ are wffs.

3. If α is a wff, so is ~α.

4. If α and β are wffs, then so are:

- (α · β)
- (α ∨ β)
- (α ⊃ β)
- (α ≡ β)

5. These are the only ways to make wffs.

The scope and binding rules are the same for **FF** as they were for **M** and **F**.

FF: Semantics

- The semantics for **FF** are basically the same as for **F**.
- We insert an interpretation of function symbols.
  - Step 1. Specify a set to serve as a domain of interpretation, or domain of quantification.
  - Step 2. Assign a member of the domain to each constant.
  - **Step 3. Assign a function with arguments and ranges in the domain to each function symbol.**
  - Step 4. Assign some set of objects in the domain to each one-place predicate; assign sets of ordered n-tuples to each relational predicate.
  - Step 5. Use the customary truth tables for the interpretation of the connectives.
- The function assigned in Step 3 will be a function in the meta-language used to interpret the function in the object language.

Translations Into FF

- Translation key:
  - Lxy: x loves y
  - f(x):the father of x
  - g(x):the mother of x
- Olaf loves his mother
  - Log(o)
- Olaf loves his grandmothers
  - Log(g(o)) • Log(f(o))
- No one is his/her own mother
  - (∀x)~x=g(x)

Functions and Mathematics

- Many simple concepts in arithmetic are functions: addition, multiplication, least common multiple.
- The most fundamental function in mathematics is the successor function.
- All other mathematical functions can be defined in terms of successor and other basic concepts.
- All of arithmetic can be developed from five basic axioms, called the Peano axioms.
- Peano's Axioms for Arithmetic

  P1: 0 is a number

  P2: The successor (x') of every number (x) is a number

  P3: 0 is not the successor of any number

  P4: If x'=y' then x=y

  P5: If P is a property that may (or may not) hold for any number, and if

  i. 0 has P; and

  ii. for any x, if x has P then x' has P;

  then all numbers have P.

Peano's Axioms, Regimented

Key: a: zero; Nx: x is a number; f(x): the successor of x

P1. Na

P2. $(\forall x)(Nx \supset Nf(x))$

P3. $\sim(\exists x)(Nx \bullet f(x)=a)$

P4. $(\forall x)(\forall y)[(Nx \bullet Ny) \supset (f(x)=f(y) \supset x=y)]$

P5. $\{Pa \bullet (\forall x)[(Nx \bullet Px) \supset Pf(x)]\} \supset (\forall x)(Nx \supset Px)$

Some Number-Theoretic Statements

- Key:
  - o: one
  - f(x): the successor of x
  - f(x, y): the product of x and y
  - Ex: x is even
  - Ox: x is odd
  - Px: x is prime

1. One is not the successor of any number.
- $(\forall x)(Nx \supset \sim f(x)=o)$

2. If the product of a pair of numbers is odd, then the product of the successors of those numbers is even.
- $(\forall x)(\forall y)\{(Nx \bullet Ny) \supset [Of(x, y) \supset Ef(f(x), f(y))]\}$

3. There are no prime numbers such that their product is prime.
- $\sim(\exists x)(\exists y)[Nx \bullet Px \bullet Ny \bullet Py \bullet Pf(x, y)]$

Derivations Using Functions

- No new rules
- Functions act like simple terms.
- A functor can be either a constant or a variable.
  - It depends on what the arguments of the function are.
- We can UI to a variable, or a function of a variable, or any complex function all of whose arguments are variables.
- For UG, if the arguments of a function are all variables, then we are free to use UG over the variables in that function.
  - If the arguments contain any constants, then we can not use UG.
- For EI, we must continue always to instantiate to a new term.
  - A functor is not a new term if any of its arguments, or any of the arguments of any of its sub-functors, have already appeared in the derivation.

The Motivating Argument

1. No odd numbers are even.
2. One is odd.
3. One is the square of one.
So, not all square numbers are even.

1. $(\forall x)(Ox \supset \sim Ex)$
2. Oo
3. $o=f(o)$

$/ \sim(\forall x)Ef(x)$

More Derivations

1. $(\forall x)[Ax \supset Bxf(x)]$
2. $(\exists x)Af(x)$ / $(\exists x)Bf(x)f(f(x))$

---

1. $\sim(\exists x)Cx$ / $(\forall x)\sim Cf(x, g(x))$

---

1. $(\forall x)\{(Nx \bullet Gxt) \supset (\exists y)(\exists z)[Py \bullet Pz \bullet x=f(y, z)]\}$
2. $Nb \bullet Gbt$ / $(\exists x)(\exists y)(\exists z)[Nx \bullet Py \bullet Pz \bullet x=f(y, z)]$