Construtivismo Matemático e suas aplicações na computação

Rafael Castro G. Silva

rafaelcgs10@gmail.com

Departamento de Ciência da Computação
Centro de Ciências e Tecnológicas
Universidade do Estado de Santa Catarina

August 30, 2017

Motivação

Uma pequena revisão de lógica clássica

O que significa afirmar que $P \vee Q$ é verdade?

O que significa afirmar que $P \vee \neg P$ é verdade?

Para toda afirmação P, ou P ou $\neg P$ é verdade.

(*Law of excluded middle*)

Uma pequena revisão de lógica clássica

O que significa afirmar que $P \vee Q$ é verdade?

O que significa afirmar que $P \vee \neg P$ é verdade?

Para toda afirmação P, ou P ou $\neg P$ é verdade.

(*Law of excluded middle*)

E quando não há prova de P ou de $\neg P$?

Todo inteiro par maior que 2 pode ser escrito como a soma de dois números primos. (*Goldach Conjecture*)

A interpretação clássica dos conectivos lógicos

Tabela verdade!

A interpretação clássica dos conectivos lógicos

Tabela verdade!

$P \vee Q$ é interpretado como $\neg(\neg P \wedge \neg Q)$:

é uma contradição P e Q serem falsos.

O que leva a interpretação idealizada (platônica) da existência:

$\exists x P(x)$ significa $\neg \forall x \neg P(x)$:

é uma contradição $P(x)$ ser falso para todo x.

A interpretação clássica dos conectivos lógicos

Tabela verdade!

$P \vee Q$ é interpretado como $\neg(\neg P \wedge \neg Q)$:

é uma contradição P e Q serem falsos.

O que leva a interpretação idealizada (platônica) da existência:

$\exists x P(x)$ significa $\neg \forall x \neg P(x)$:

é uma contradição $P(x)$ ser falso para todo x.

e assim se fez toda a matemática.

Sistema Formal

- Conjunto de símbolos.
- Uma gramática, que define termos bem definidos a partir dos símbolos.
- Aparato de dedução: regras que determinam algum tipo de consequência entre os termos bem formados.
- Semântica para os termos.

Programa de Hilbert

- Formalizar a matemática: escrever toda a matemática como um sistema formal.
- Completude: Provar que todas as verdades (e falsidades) matemáticas podem ser provadas nesse sistema formal.
- Consistência: Provar que uma verdade jamais deriva uma falsidade.
- Decibilidade: Um algoritmo que seja capaz provar todas as verdades e falsidades.

Primeiro Teorema da Incompletude

- A Aritmética de Peano é incompleta, ou seja, existe um fórmula φ indecidível tal que:
  $PA \nvdash \varphi$ e $PA \nvdash \neg\varphi$
- A ideia da prova consiste na construção de uma frase com auto-referência, como paradoxo do mentiroso:
  "Esta frase é falsa"
- Em analogia, objetiva-se construir uma frase (ou proposição) em PA que diga:
  "Esta proposição e sua negação não podem ser provadas"

Curry-Howard

Curry-Howard ... Cálculo Lambda Tipado ... Dedução Natural

Construtivismo

Lógica clássica VS Lógica construtivista

Law of Excluded Middle: $\forall P, P \vee \neg P$ (LEM)

E quando não há prova de P e não há prova de $\neg P$ o que podemos dizer sobre $P \vee \neg P$?

- Clássica: Verdade
- Construtivista: Não provável

Na lógica clássica dizemos que proposições são verdadeiras ou falsas. Na construtivista dizemos que proposições são prováveis ou não prováveis.

Construtivismo é sobre não aceitar LEM.

Intuicionismo

A interpretação intuicionista dos conectivos lógicos

Interpretação BHK (Brouwer, Heyting e Kolmogorv):

- $\vee$ - para provar $P \vee Q$ é necessário ter uma prova de P ou uma prova de Q.
- $\wedge$ - para provar $P \wedge Q$ é necessário ter uma prova de P e uma prova de Q.
- $\rightarrow$ - para provar $P \rightarrow Q$ é necessário ter uma algoritmo que converte uma prova de P em uma prova de Q.
- $\neg$ - para provar $\neg P$ é necessário mostrar que P implica numa contradição $(0 = 1)$.
- $\exists$ - para provar $\exists x P(x)$ é necessário ter uma construção de um objeto x e provar que $P(x)$ é verdade.
- $\forall$ - para provar $\forall x \in S\ P(x)$ é necessário ter um algoritmo que aplicado a qualquer objeto x e a prova de que $x \in S$, prova que $P(x)$ é verdade.

Sistemas de Tipos

Qual é a graça de sistemas de tipos?

Sistemas de Tipos

Qual é a graça de sistemas de tipos?

- GRAÇA 1: polimorfismo.

```
len :: [a] -> Int
len [] = 0
len (x:xs) = 1 + len xs
```

Sistemas de Tipos

Qual é a graça de sistemas de tipos?

- GRAÇA 1: polimorfismo.

```
len :: [a] -> Int
len [] = 0
len (x:xs) = 1 + len xs
```

- GRAÇA 2: *Lightweight Formal Methods*.

Coq

Ferramentas que auxiliam o desenvolvimento de provas formais.
Verificam a consistência lógica de uma prova matemática escrita em linguagem.