Lambda Calculus in a hurry

Rafael Castro G. Silva

rasi@di.ku.dk

Department of Computer Science
University of Copenhagen

29/06/2022

Motivation

- Lambda Calculus is very important in Computer Science!
- Its has many applications in Computer Science
  - Do you know any?
- No prerequisites to learn it (maybe just Set Theory)
- It sounds really cool

Motivation

- Lambda Calculus is very important in Computer Science!
- Its has many applications in Computer Science
  - Do you know any?
- No prerequisites to learn it (maybe just Set Theory)
- It sounds really cool

Motivation

- Lambda Calculus is very important in Computer Science!
- Its has many applications in Computer Science
  - Do you know any?
- No prerequisites to learn it (maybe just Set Theory)
- It sounds really cool

Motivation

- Lambda Calculus is very important in Computer Science!
- Its has many applications in Computer Science
  - Do you know any?
- No prerequisites to learn it (maybe just Set Theory)
- It sounds really cool

Contents

- A bit of history
- What is and is not a lambda term?
- How do we compute?
- What cool things can we do with it?

Learning Outcomes

- Learn some background history about Logic and Computer Science
- Identify well-formed terms in Lambda Calculus
- Apply beta-reduction to terms
- Write down/design your own programs in Lambda Calculus

A bit of history

Set Theory

- A set is a collection of elements
  - Unique elements
  - The order of the elements in the set doesn't matter
  - Examples:
    - Numbers
    - We in this room
    - The set of all sets
- A very simple foundation for mathematics

Figure: George Cantor

Russel's Paradox: Set Theory is broken!

Let $R = \{x \mid x \notin x\}$, then $R \in R \iff R \notin R$

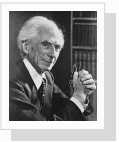

Figure: Bertrand Russel

Set Theory

- A set is a collection of elements
  - Unique elements
  - The order of the elements in the set doesn't matter
  - Examples:
    - Numbers
    - We in this room
    - The set of all sets
- A very simple foundation for mathematics

Figure: George Cantor

Russel's Paradox: Set Theory is broken!

Let $R = \{x \mid x \notin x\}$, then $R \in R \iff R \notin R$

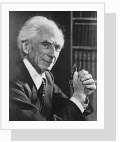

Figure: Bertrand Russel

Set Theory

- A set is a collection of elements
  - Unique elements
  - The order of the elements in the set doesn't matter
  - Examples:
    - Numbers
    - We in this room
    - The set of all sets
- A very simple foundation for mathematics

Figure: George Cantor

Russel's Paradox: Set Theory is broken!

Let $R = \{x \mid x \notin x\}$, then $R \in R \iff R \notin R$

Figure: Bertrand Russel

The Problem

You are David Hilbert and Mathematics is broken

Hilbert's Program

- Provide a new foundation form mathematics
  - The system must be powerful enough to do arithmetic
  - Provide a proof that the system is consistent
  - **Provide an algorithm that solves any given problem stated in this system**
- What do we need here?
  - What is an algorithm?

Figure: David Hilbert

The Problem

The Problem

You are David Hilbert and Mathematics is broken

Hilbert's Program

- Provide a new foundation form mathematics
  - The system must be powerful enough to do arithmetic
  - Provide a proof that the system is consistent
  - **Provide an algorithm that solves any given problem stated in this system**
- What do we need here?
  - What is an algorithm?

Figure: David Hilbert

The Problem

The Problem

You are David Hilbert and Mathematics is broken

Hilbert's Program

- Provide a new foundation form mathematics
  - The system must be powerful enough to do arithmetic
  - Provide a proof that the system is consistent
  - **Provide an algorithm that solves any given problem stated in this system**
- What do we need here?
  - What is an algorithm?

Figure: David Hilbert

The Problem

The Problem

You are David Hilbert and Mathematics is broken

Hilbert's Program

- Provide a new foundation form mathematics
  - The system must be powerful enough to do arithmetic
  - Provide a proof that the system is consistent
  - **Provide an algorithm that solves any given problem stated in this system**
- What do we need here?
  - What is an algorithm?

Figure: David Hilbert

The Formal Definition of Algorithm

- Three formal definitions of algorithm showed up together
  - **Alonzo Church: Lambda Calculus in 1935**
    - It's a language

  - Kurt Gödel: Recursive Functions in 1935
    - It's a class of functions

  - Alan Turing: Turing Machines in 1936
    - It's an abstract machine

What is and is not a lambda term?

The Syntax of Lambda Calculus

Given a set of variables V, a λ-term is inductively defined by:

- All variables: $x, y, z, w... \in V$ (atoms) are λ-terms
- If M is a λ-term and x is a variable, then $\lambda x.M$ is a λ-term. (λ abstraction or λ function)
- If M and N are λ-terms, then (MN) is a λ-term. (application)

Parenthesis omission convention: $(MN)P \equiv MNP$ for any given M, N and P.

Examples

Considering $V \equiv \{x, y, z\}$,

1. x
2. y
3. z
4. $\lambda x.x$
5. $\lambda x.y$
6. $(\lambda x.x)y$
7. $(\lambda x.x)(\lambda x.x)$

The Syntax of Lambda Calculus

Given a set of variables V, a λ-term is inductively defined by:

- All variables: $x, y, z, w... \in V$ (atoms) are λ-terms
- If M is a λ-term and x is a variable, then $\lambda x.M$ is a λ-term. (λ abstraction or λ function)
- If M and N are λ-terms, then (MN) is a λ-term. (application)

Parenthesis omission convention: $(MN)P \equiv MNP$ for any given M, N and P.

Examples

Considering $V \equiv \{x, y, z\}$,

1. x
2. y
3. z
4. $\lambda x.x$
5. $\lambda x.y$
6. $(\lambda x.x)y$
7. $(\lambda x.x)(\lambda x.x)$

Exercise 1 (Individual and in 5 min)

Which of these are well-formed λ-terms?

Considering $V \equiv \{x, y\}$,

1. x
2. $x(\lambda x.x)$
3. MNP
4. $\lambda x.y$
5. $\lambda x.\lambda y.xy$
6. z
7. $(\lambda y.y)(\lambda y.yx)$

Syntax definition

Given a set of variables V, a λ-term is inductively defined by:

- All variables: $x, y, z, w... \in V$ (atoms) are λ-terms
- If M is a λ-term and x is a variable, then $\lambda x.M$ is a λ-term. (λ abstraction or λ function)
- If M and N are λ-terms, then (MN) is a λ-term. (application)

Parenthesis omission convention:
$(MN)P \equiv MNP$ for any given M, N and P.

How do we compute?

Free variables

The set of all free variables of M, called $FV(M)$, is defined inductively by:

$$\begin{aligned} FV(x) &= \{x\} \\ FV(MN) &= FV(M) \cup FV(N) \\ FV(\lambda x.M) &= FV(M) \setminus \{x\} \end{aligned}$$

Examples

- $FV(x) = \{x\}x$
- $FV(xy) = \{x, y\}$
- $FV(\lambda x.x) = \emptyset$
- $FV(\lambda x.xy) = \{y\}$

Beta Reduction

Substitution

- $[N/x]x \equiv N$;
- $[N/x]y \equiv y$, if $x \not\equiv y$;
- $[N/x](PQ) \equiv ([N/x]P[N/x]Q)$;
- $[N/x](\lambda x.P) \equiv \lambda x.P$;
- $[N/x](\lambda y.P) \equiv \lambda y.P$, if $x \notin FV(P)$;
- $[N/x](\lambda y.P) \equiv \lambda y.[N/x]P$, if $x \in FV(P)$ and $y \notin FV(N)$;
- $[N/x](\lambda y.P) \equiv \lambda z.[N/x][z/y]P$, if $x \in FV(P)$: and $y \in FV(N)$;

Beta Reduction

A term of the form $(\lambda x.M)N$ is called a redex β and is interpreted by the substitution $[N/x]M$

Beta Reduction

Substitution

- $[N/x]x \equiv N$;
- $[N/x]y \equiv y$, if $x \not\equiv y$;
- $[N/x](PQ) \equiv ([N/x]P[N/x]Q)$;
- $[N/x](\lambda x.P) \equiv \lambda x.P$;
- $[N/x](\lambda y.P) \equiv \lambda y.P$, if $x \notin FV(P)$;
- $[N/x](\lambda y.P) \equiv \lambda y.[N/x]P$, if $x \in FV(P)$ and $y \notin FV(N)$;
- $[N/x](\lambda y.P) \equiv \lambda z.[N/x][z/y]P$, if $x \in FV(P)$: and $y \in FV(N)$;

Beta Reduction

A term of the form $(\lambda x.M)N$ is called a redex β and is interpreted by the substitution $[N/x]M$

Beta Reduction Examples

Examples

- $(\lambda x.x)y$
- $(\lambda x.x)(\lambda x.x)$
- $(\lambda x.\lambda y.y)y$
- $(\lambda x.\lambda y.xy)y$

Beta Reduction Exercise (group and in 5 minutes)

Exercise

- $((\lambda x.x)(\lambda x.x))(\lambda x.x)$
- $(\lambda x.xx)(\lambda x.xx)$

What cool things can we do with it?

Let's program

open: www.haskell.org