Introduction to Program Synthesis (SS 2025)

Chapter 1 - Introduction

Dr. rer. nat. Roman Kalkreuth

Chair for AI Methodology (**AIM**), Department of Computer Science,
RWTH Aachen University, Germany

- Calculus (dt. Kalkül) $\rightarrow$ formal system of rules
- Used to derive statements from given statements (axioms)
- A calculus consists of
  - $\rightsquigarrow$ **Building blocks**
    - Alphabet $\rightarrow$ Symbols, Connectives, structuring signs, ...
  - $\rightsquigarrow$ **Formation rules**
    - Define how building blocks form complex objects or well-formed formulas
    - Analogy to natural language $\rightarrow$ "grammar" of the calculus
  - $\rightsquigarrow$ **Transformation rules**
    - Derivation and deduction rules
    - Transformation to create new objects from them
  - $\rightsquigarrow$ **Axioms**
    - Objects or Expressions
    - Formed according to the formation rules of the calculus
    - Obvious principle

Example (Chess Calculus)

Chess game with pieces (axioms) and moves (transformation rules).

- **Formal framework** that can be used in mathematics, logic and programming
- Approach to **systematic solving** problems in certain domain
- Design of suitable **logical frameworks** for **programming languages**
- Examples:
  - **Mathematical** $\rightarrow$ arithmetics, O-Calculus, Stochastic calculus
  - **Logic** $\rightarrow$ propositional calculus
  - **Computation** $\rightarrow$ λ-Calculus, Turing machine, Plankalkül

- λ-Calculus $\rightarrow$ minimal model of computation
  - Smallest universal programming language
  - Any computable function can be expressed and evaluated
- Developed by Alonzo Church in the 1930's
  - Published in 1941[Chu85]
- Study of functional computing
- Introduction of a functional notation: $\lambda x.y$

  $\rightsquigarrow$ Contemporary notation analogy: $x \mapsto y$

  $\rightsquigarrow$ *Formalisation* of mathematical functions
  - f(x) $= x^2 \quad \rightsquigarrow \quad x \mapsto x^2$, $\lambda x.x^2$
  - f(x,y) $= x^2 + y^2 \quad \rightsquigarrow \quad (x, y) \mapsto x^2 + y^2$, $\lambda x.\lambda y.x^2 + y^2$

- Functions are considered expressions E
  - Parenthesis can be used for clarity $E \Leftrightarrow (E)$
  - Keywords are only λ and the dot
- **Function creation:** Function denotation that has a formal argument x and a functional body $E \rightarrow \lambda x.E$
- **Function application:** Denotation of the application of a function E_1 to the argument $E_2 \rightarrow E_1.E_2$
- Syntax of Lambda Calculus:

```
<expression> := <name> | <function> | <application>
<function> := λ <name>.<expression>
<application> := <expression><expression>
```

- **Abstraction** $\rightarrow$ anonymous functions
- Single transformation rule $\rightarrow$ **variable substitution**
- Single function definition scheme $\rightarrow \lambda x.x$
  - $\rightsquigarrow$ λ symbol $\rightarrow$ start of a function expression
  - $\rightsquigarrow$ Name after $\lambda \rightarrow$ identifier of the function argument
  - $\rightsquigarrow$ Expression after the point $\rightarrow$ function body
- Functions can be applied to expressions $\rightarrow (\lambda x.x)y$
  - $\rightsquigarrow$ Evaluation $\rightarrow$ substitution of the argument x
  - $\rightsquigarrow$ $(\lambda x.x)y = [y/x]\,x = y$
  - $\rightsquigarrow$ $[y/x] \rightarrow$ Notation to indicate the substitution of x by y

- Variables can be either free or bound like in math
  - $\rightsquigarrow$ **Free variable:** Symbol in an expression that can be substituted
  - $\rightsquigarrow$ **Bound variable:** Symbol bound to logical quantifiers or variable-binding operators:
    $\sum_{x=0}^{N}$ $\prod_{x=0}^{\infty}$ $\forall x$ $\exists x$
- $(\lambda x.xy) \rightarrow$ variable x is bound and y is free
- $(\lambda x.x)(\lambda y.yx)$
  - $\rightsquigarrow$ x in the first expression is bound to the first λ
  - $\rightsquigarrow$ y in the second expression is bound to the second λ
  - $\rightsquigarrow$ x in the second expression is free

- Functions in standard λ calculus are anonymous
  - $\rightsquigarrow$ However, capital letters are commonly used to simplify the notation
- For instance, the identity function denoted with `I` serves as a synonym for $(\lambda x.x)$
- **Substitution:** Fundamental mechanism in λ calculus
- Computational approach to function composition:
  $g(x) := (u \circ v)(x) = u(v(x))$

Example (Identity function)

- We apply the identity function to itself which is an application:
  $\rightsquigarrow$ $\mathtt{II} \equiv I_1.I_2 \equiv (\lambda x.x)(\lambda x.x)$
- We can rewrite the expression as:
  - $\mathtt{II} \equiv (\lambda x.x)(\lambda z.z)$
- The identity function when applied to itself leads therefore to:
  - $\mathtt{II} \equiv (\lambda x.x)(\lambda z.z) = [\lambda z.z/x]\, x = \lambda z.z \equiv \mathtt{I}$

References I

[Chu85] Alonzo Church. *The Calculi of Lambda-Conversion*. Princeton: Princeton University Press, 1985. ISBN: 9781400881932. DOI: doi:10.1515/9781400881932. URL: https://doi.org/10.1515/9781400881932.