On the Algebraic Definition of Programming Languages

Ambrus Kaposi[1]

[1]Eötvös Loránd University

8 May 2020, online

PROGRAM
FINANCED FROM
THE NRDI FUND

Software crisis

Complex systems, many bugs.

How do we improve **existing software**?

Develop tools to

- test code (Attila Kovács)
- analyse code (Melinda Tóth, István Bozó)
- refactor code in a correct way (Dániel Horpácsi)

National Research, Development and Innovation Office Hungary

PROGRAM FINANCED FROM THE NRDI FUND

Software crisis

Complex systems, many bugs.

How do we improve **future software**?

Develop new programming languages

- it should be easy to write correct code
- hard to write incorrect code

National Research, Development and Innovation Office
Hungary

PROGRAM FINANCED FROM THE NRDI FUND

Why do we need different programming languages?

Programming languages are the same:

- They are just different ways to express CPU instructions.
- They are all Turing-complete.

National Research, Development and Innovation Office
Hungary

PROGRAM FINANCED FROM THE NRDI FUND

Why do we need different programming languages?

Programming languages are the same:

- They are just different ways to express CPU instructions.
- They are all Turing-complete.

They are not the same:

- If the language suits the problem, we can express the solution **directly**.
- If not, we need to **encode** the solution.

NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION OFFICE HUNGARY
PROGRAM FINANCED FROM THE NRDI FUND

Programming is the process of encoding

Domain specific languages:

- HTML for hypertext
- P4 for programming switches (Sándor Laki)
- Erlang for programming mobile networks, distributed systems
- Alan Kay's example:
  - operating system with GUI, web browser, spreadsheet, word processor
  - all in 20thousand lines of code

National Research, Development and Innovation Office
Hungary
PROGRAM FINANCED FROM THE NRDI FUND

Difficulties of creating new languages

- Design the language such that it is sound
  - Prove its properties
- Implementation, lots of technical details
  - Improvements recently, e.g. LLVM
- Interface to other languages

People usually don't bother: instead, they follow the hammer principle.

National Research, Development and Innovation Office
Hungary
PROGRAM FINANCED FROM THE NRDI FUND

What is a domain specific language for describing languages?

This is our research project.

What is a language?

A language is a collection of programs.

What is a program?

National Research, Development and Innovation Office
Hungary

PROGRAM FINANCED FROM THE NRDI FUND

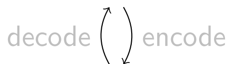
What is a program?

(1) number

National Research, Development and Innovation Office
Hungary
PROGRAM FINANCED FROM THE NRDI FUND

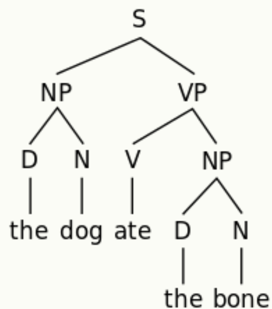
What is a program?

```
icationInstanceManager.setApplicationInstanceListener(new ApplicationInstanceListener() {
@Override
public void newInstanceCreated(String[] args) {
    System.out.println("New instance detected, args length=" + args.length);
    if (args.length > 1)
        if (args[0].equals("-open")) {
            if (facsKinView.getFileChooser().isVisible())
                facsKinView.getFileChooser().cancelSelection();
            facsKinView.openFile(args[1]);
            //TODO: nem szabadna, hogy ilyen kelljen:
            //waiting 1 second
            try { Thread.sleep(1000); } catch (InterruptedException ex) {}
            //long t0, t1;t0 = System.currentTimeMillis(); do { t1 = System.currentTimeMi
            for (int i = 2; i < args.length; i ++) {
                facsKinView.openFile(args[i]);
            }
        }
}
```



National Research, Development and Innovation Office
Hungary
PROGRAM FINANCED FROM THE NRDI FUND

What is a program?



(3) syntax tree

parsing ↑ ↓ print

(2) string

decode ↑ ↓ encode

(1) number

NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION OFFICE HUNGARY

PROGRAM FINANCED FROM THE NRDI FUND

What is a program?

(4) syntax tree with bindings

scope checking ↑↓ pick variable names

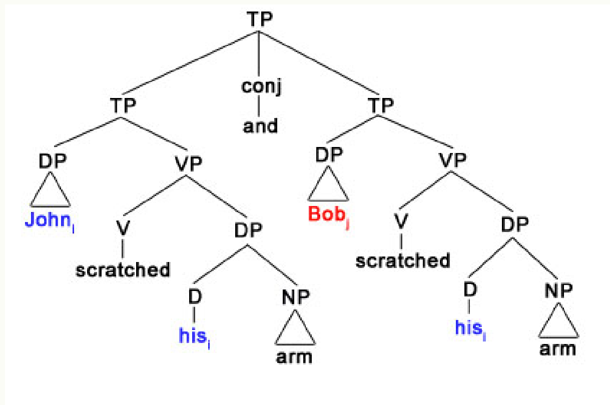
(3) syntax tree

parsing ↑↓ print

(2) string

decode ↑↓ encode

(1) number



NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION OFFICE HUNGARY

PROGRAM FINANCED FROM THE NRDI FUND

What is a program?

(5) well typed syntax

type checking ⟲

(4) syntax tree with bindings

scope checking ⟲ pick variable names

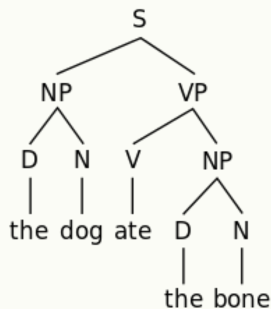
(3) syntax tree

parsing ⟲ print

(2) string

decode ⟲ encode

(1) number



NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION OFFICE HUNGARY
PROGRAM FINANCED FROM THE NRDI FUND

What is a program?

(5) well typed syntax

type checking

(4) syntax tree with bindings

scope checking | pick variable names

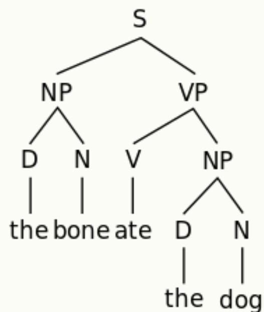
(3) syntax tree

parsing | print

(2) string

decode | encode

(1) number



NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION OFFICE HUNGARY
PROGRAM FINANCED FROM THE NRDI FUND

What is a program?

(5) well typed syntax

type checking

(4) syntax tree with bindings

scope checking | pick variable names

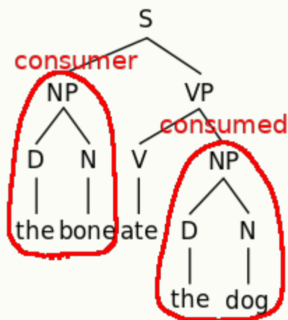
(3) syntax tree

parsing | print

(2) string

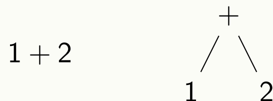
decode | encode

(1) number



National Research, Development and Innovation Office Hungary
PROGRAM FINANCED FROM THE NRDI FUND

What is a program?



(5) well typed syntax

type checking

(4) syntax tree with bindings

scope checking | pick variable names

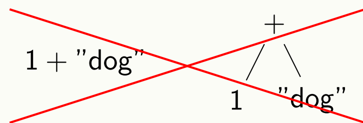
(3) syntax tree

parsing | print

(2) string

decode | encode

(1) number



National Research, Development and Innovation Office
Hungary
PROGRAM FINANCED FROM THE NRDI FUND

What is a program?

(6) well typed syntax quotiented

normalise

(5) well typed syntax

type checking

(4) syntax tree with bindings

scope checking | pick variable names

(3) syntax tree

parsing | print

(2) string

decode | encode

(1) number

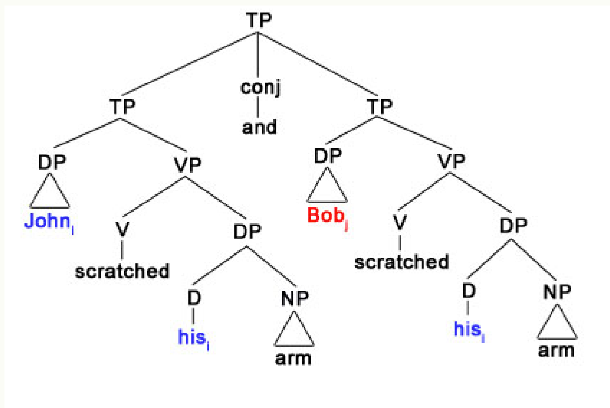
$1 + 2 = 3$

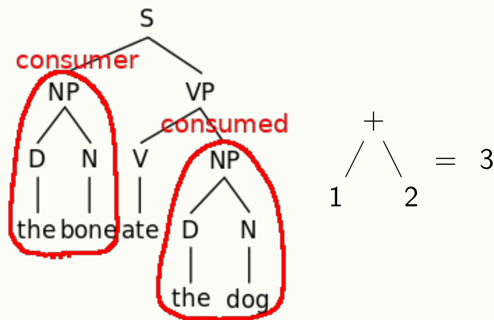
+
1 2
$=$ 3

National Research, Development and Innovation Office
Hungary

PROGRAM FINANCED FROM THE NRDI FUND

A language is a collection of:





Mathematicians call this a **generalised algebraic structure**, programmers call it a **quotient inductive-inductive type**.

NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION OFFICE
HUNGARY
PROGRAM FINANCED FROM THE NRDI FUND

Results

- We created the first programming language which supports quotient inductive-inductive types (QIIT).
  - It is based on Martin-Löf's type theory and we call it setoid type theory.
  - Syntax (as a QIIT) and constructive semantics.
  - A prototype implementation in Haskell.
- We developed the metatheory of QIITs.
  - A concrete description.
  - We proved their consistency, constructive semantics.
- We developed techniques to formalise metatheoretic results in existing languages based on type theory.

NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION OFFICE HUNGARY
PROGRAM FINANCED FROM THE NRDI FUND

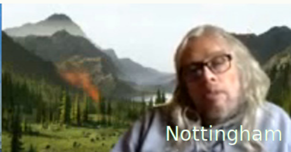
Further goals

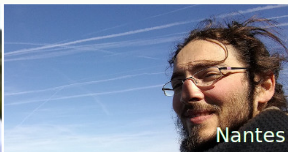
- Theoretical: show decidability of type checking for setoid type theory
- Theoretical: reduction of QIITs to basic combinators
- Practical: industrial strength implementation
- Practical: implement existing programming languages as QIITs
- Long term: implement setoid type theory in itself

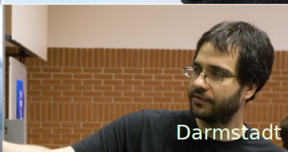
National Research, Development and Innovation Office
Hungary

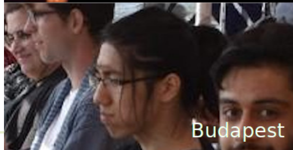
PROGRAM FINANCED FROM THE NRDI FUND

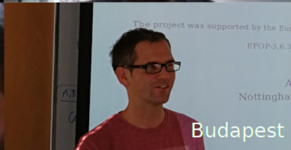
Team and collaborators

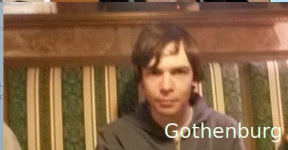












PROGRAM
FINANCED FROM
THE NRDI FUND

Summary

- A good programming language admits direct representations of the domain
- Programming languages are usually **encoded** instead of **directly** represented
- Our hypothesis: the reason for this is that there is no programming language supporting QIITs
- We are developing such a language: the theoretical foundations are mostly done
- We would like to develop a practical implementation

NATIONAL RESEARCH, DEVELOPMENT AND INNOVATION OFFICE
HUNGARY

PROGRAM FINANCED FROM THE NRDI FUND

Q&A

PROGRAM
FINANCED FROM
THE NRDI FUND

References

- Yoshiki Ohshima, Ted Kaehler Dan Amelang, Bert Freudenberg, Aran Lunzer, Ian Piumarta Alan Kay, Takashi Yamamiya, Hesam Samimi Alan Borning, Bret Victor, and Kim Rose, Steps toward the reinvention of programming, 2012 final report. Submitted to the National Science Foundation (NSF).
- Thorsten Altenkirch and Ambrus Kaposi. 2016. Type theory in type theory using quotient inductive types. SIGPLAN Not. 51, 1 (January 2016), 18–29.
- Ambrus Kaposi, András Kovács, and Thorsten Altenkirch. 2019. Constructing quotient inductive-inductive types. Proc. ACM Program. Lang. 3, POPL, Article 2 (January 2019), 24 pages.
- Altenkirch T., Boulier S., Kaposi A., Tabareau N. (2019) Setoid Type Theory—A Syntactic Translation. In: Hutton G. (eds) Mathematics of Program Construction. MPC 2019. Lecture Notes in Computer Science, vol 11825. Springer, Cham
- András Kovács, Ambrus Kaposi. Large and Infinitary Quotient Inductive-Inductive Types. Thirty-Fifth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS). 8–11 July 2020
- Ambrus Kaposi, Jakob von Raumer. A Syntax for Mutual Inductive Families. To appear at the 5th International Conference on Formal Structures for Computation and Deduction (FSCD 2020). June 29–July 6, 2020
- Ambrus Kaposi, András Kovács and Ambroise Lafont. For induction-induction, induction is enough with. Submitted to TYPES 2019 post-proceedings