UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Jevgeni Savostkin

Improving accuracy of BCI with multiple simultaneous users

Master's Thesis (30 ECTS)

Supervisor: Ilya Kuzovkin, MSc
Supervisor: Raul Vicente, PhD

Tartu 2017

Improving accuracy of BCI with multiple simultaneous users

Abstract: Many interpreting program languages are dynamically typed, such as Visual Basic or Python. As a result, it is easy to write programs that crash due to mismatches of provided and expected data types. One possible solution to this problem is automatic type derivation during compilation. In this work, we consider study how to detect type errors in the WHITESPACE language by using fourth order logic formulae as annotations. The main result of this thesis is a new triple-exponential type inference algorithm for the fourth order logic formulae. This is a significant advancement as the question whether there exists such an algorithm was an open question. All previous attempts to solve the problem lead lead to logical inconsistencies or required tedious user interaction in terms of interpretative dance. Although the resulting algorithm is slightly inefficient, it can be used to detect obscure programming bugs in the WHITESPACE language. The latter significantly improves productivity. Our practical experiments showed that productivity is comparable to average Java programmer. From a theoretical viewpoint, the result is only a small advancement in rigorous treatment of higher order logic formulae. The results obtained by us do not generalise to formulae with the fifth or higher order.

Keywords:

List of keywords

CERCS:

CERCS code and name: `https://www.etis.ee/Portal/Classifiers/Details/d3717f7b-bec8-4cd9-8ea4-c89cd56ca46e`

Contents

1 Introduction **4**

2 Related work review **5**
2.1 Title of Subsection 1 5

3 Technical background **7**

4 Other Ways to Represent Data **8**
4.1 Tables 8
4.2 Lists 8
4.3 Math mode 8
4.4 algorithm2e 9
4.5 Pseudocode 9
4.6 Frame Around Information 9

5 Conclusion **10**

6 Eestikeelne pealkiri **11**

1 Introduction

Tip: if it's hard for you to start writing, then try to split it to smaller parts, e.g. if the title is "Type Inference for a Cryptographic Protocol Prover Tool" then the "What is it" can be divided into "what is type inference", "what is cryptographic protocol" and "what is the prover tool". These three can also be split to smaller parts etc.

2 Related work review

Short description of what this section is about

2.1 Title of Subsection 1

Cross-references to figures, tables and other document elements. LaTeX internally numbers all kind of objects that have sequence numbers:

- chapters, sections, subsections;
- figures, tables, algorithms;
- equations, equation arrays.

To reference them automatically, you have to generate a label using `\label{some-name}` just after the object that has the number inside. Usually, labels of different objects are split into different namespaces by adding dedicated prefix, such as `sec:`, `fig:`. To use the corresponding reference, you must use command `\ref` or `\eqref`. For instance, we can reference this subsection by calling Section **??**. Note that there should be a nonbreakable space ˜ between the name of the object and the reference so that they would not appear on different lines.

Citations. Usually, you also want to reference articles, webpages, tools or programs or books. For that you should use citations and references. The system is similar to the cross-referencing system in LaTeX. For each reference you must assign a unique label. Again, there are many naming schemes for labels. However, as you have a short document anything works. To reference to a particular source you must use `\cite{label}` or `\cite[page]{label}`.

References themselves can be part of a LaTeX source file. For that you need to define a bibliography section. However, this approach is really uncommon. It is much more easier to use BibTeX to synthesise the right reference form for you. For that you must use two commands in the LaTeX source

- \bibliographystyle{alpha} or \bibliographystyle{plain}
- \bibliography{file-name}

The first command determines whether the references are numbered by letter-number combinations or by cryptic numbers. It is more common to use `alpha` style. The second command determines the file containing the bibliographic entries. The file should end with `bib` extension. Each reference there is in specific form. The simplest way to avoid all technicalities is to use graphical frontend Jabref (`http:`

//jabref.sourceforge.net/) to manage references. Another alternative is to use DBLP database of references and copy BibTeX entries directly form there.

The following paragraph shows how references can be used. Game-based proving is a way to analyse security of a cryptographic protocol [BR04, Sho04]. There are automatic provers, such as CertiCrypt [BGZ09] and ProVerif [Bla].

3 Technical background

Here are a few examples of how to add figures or pictures to your thesis (see Figures **??**, **??**, **??**).

Rule: All the figures, tables and extras in the thesis have to be referred to somewhere in the text.

Tip: If you add a screenshot then labeling the parts might help make the text more understandable (panel C vs bottom left part), e.g.

Example: A screenshot of ProveIt can be seen on Figure **??**. The user first enters the pseudocode of the initial game in panel B. ProveIt also keeps track of all the previous games showing the progress on a graph seen in panel A.

4 Other Ways to Represent Data

4.1 Tables

Statement	**Typeset Example**
assignment	$a := 5 + b$
uniform choice	$m \leftarrow M$
function signature	$f : K \times M \rightarrow L$

Table 1: Statements in the ProveIt language

4.2 Lists

Numbered list example:

1. item one;
2. item two;
3. item three.

4.3 Math mode

Example:

$$a + b = c + d$$

Aligning:

$$\begin{aligned} a &= 5 \\ b + c &= a \\ a - 2 * 3 &= 5/4 \end{aligned}$$

Hint: Variables or equations in text are separated with $ sign, e.g. a, $x - y$.

Inference Rules

$$\text{addition}\,\frac{\Gamma \vdash x : T \qquad \Gamma \vdash y : T}{\Gamma \vdash x + y : T}$$

Bigger example:

$$\text{assign}\,\frac{\Gamma \vdash c := a + b \qquad \text{addG}\,\dfrac{\Gamma \vdash a : \mathsf{Rat} \qquad \text{var}\,\dfrac{\Gamma \vdash b : \mathsf{Int} \qquad \Gamma \vdash \mathsf{Int} \subseteq \mathsf{Rat}}{\Gamma \vdash b : \mathsf{Rat}}}{\Gamma \vdash a + b : \mathsf{Rat}}}{\Gamma \vdash c : \mathsf{Rat}}$$

4.4 algorithm2e

Algorithm 1: typeChecking

```
Input: Abstract syntax tree
Result: Type checking result; In addition, type table $\mathsf{type}_{\mathsf{type\_G}}$ for global
        variables, $\mathsf{type}_{\mathsf{game}}$ for the main game and $\mathsf{type}_{\mathsf{fun}}$ for each $fun \in F$
1 while something changed in last cycle do
2     foreach global statement s do parseStatement(s, $\mathsf{type}_{\mathsf{type\_G}}$);
3     ;
4     foreach function fun do
5         foreach statement s in fun do parseStatement(s, $\mathsf{type}_{\mathsf{fun}}$);
6         ;
7     foreach statement s in game do parseStatement(s, $\mathsf{type}_{\mathsf{game}}$);
8     ;
```

4.5 Pseudocode

```
expression
  : NUMBER
  | VARIABLE
  | '+' expression
  | expression '+' expression
  | expression '*' expression
  | function_name '(' parameters ')'
  | '(' expression ')'
```

Figure 1: Grammar of arithmetic expressions

4.6 Frame Around Information

Tip: We can use minipage to create a frame around some important information.

1. integer division (\div) - only usable between Int types
2. remainder (%) - only usable between Int types

Figure 2: Arithmetic operations in ProveIt revisited

5 Conclusion

what did you do?

What are the results?

future work?

6 Eestikeelne pealkiri

BakalaureusetÃűÃű(6 EAP)
Eesnimi Perekonnanimi
ResÃijmee

Use introduction and conclusion to give a brief overview of what this thesis is about

References

[BGZ09] Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. Formal certification of code-based cryptographic proofs. In *36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009*, pages 90–101. ACM, 2009.

[Bla] Bruno Blanchet. Proverif: Cryptographic protocol verifier in the formal model. `http://www.proverif.ens.fr/`.

[BR04] Mihir Bellare and Phillip Rogaway. Code-based game-playing proofs and the security of triple encryption. Cryptology ePrint Archive, Report 2004/331, 2004. `http://eprint.iacr.org/`.

[Kam12] Liina Kamm. ProveIt - How to make proving cryptographic protocols less tedious. Talk at the 21st Estonian Computer Science Theory Days at Kubija, January 2012.

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. `http://eprint.iacr.org/`.